PSEUDO-GIANT: A SHARING-DELINKING PARADIGM FOR EFFICIENT MULTI-TRILLION PARAMETER PRE-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent expeditious developments in deep learning algorithms, distributed training, and even hardware design for large models have enabled the training of extreme-scale models, say GPT-3 and Switch Transformer that possess hundreds of billions or even trillions of parameters. However, under limited resources, extreme-scale model training that that requires enormous amounts of computes and memory footprint suffers from frustratingly low efficiency in model convergence. In this paper, we propose a simple strategy called "Pseudo-to-Real" that effectively enhances the training efficiency in model convergence of high-memory-required large models while sufficiently managing GPU/CPU memory. The strategy provides a two-stage training that trains a model that contains much fewer parameters through parameter sharing with high efficiency and then removes the sharing and continues training until convergence. Pseudo-Giant is compatible with large models with architecture of sequential layers, including BERT, GPT, or M6. We conduct pretraining and finetuning experiments on M6 with 1B parameters to validate the effectiveness of Pseudo-Giant. Then we demonstrate a successful practice of the proposed method in pretraining a 10-trillion-parameter model on solely 512 GPUs in 10 days. The extremely simple strategy facilitate fast training of extremely large models with limited resources, resulting in smaller carbon footprint and contributing to greener AI.

1 INTRODUCTION

Recent years have luckily witnessed the expeditious growing of foundation models on broad data that highly contributes to the development of the whole deep learning and artificial intelligence community. Foundation models with self-supervised learning on big data has become an emerging paradigm of artificial intelligence systems [5], as they mostly possess high transferability to a wide range of downstream tasks and even multiple modalities. There is a salient tendency in foundation model evolution that the scale of foundation models across fields including natural language processing, computer vision, and cross-modality representation learning all have been growing tremendously from millions to trillions of parameters [11; 24; 25; 26; 33; 35; 6; 29; 17; 12; 19; 48; 47; 42; 45; 36] thanks to the concurrent advancement in distributed training framework [21; 27; 31; 28; 3; 38; 30] and hardware design, and these studies have made a demonstration of the neural scaling law [13]. However, the training of these transformer-based models incurs high financial costs and even environmental damage due to the massive carbon footprint, and thus training extreme-scale model under a decent amount of resources but with high efficiency should be a fundamental goal for both the research and industrial communities to achieve, which promotes the progress of greener AI [23; 34].

Generally there are two tracks of research in large-scale pretraining, dense models and sparse expert models respectively. Scaling dense models mostly increases model width and depth. An extreme case is GPT-3 [6] in 2020, a 175-billion-parameter transformer model trained with 10000 GPUs for months incurring striking financial and environmental costs. Elegant solutions include effective management of memory with gradient and optimizer state partitioning [27] or more efficient model parallelism and pipeline parallelism [35; 21; 38]. A series of following studies apply those techniques to realize fast training of 10-billion-parameter transformers with hundreds of GPUs in $1-2$ months. [19; 47; 45; 36] Sparse expert models with large model capacity are capable of fast training owing to the combination

of data parallelism and expert parallelism, and it is even able to train a 1-trillion-parameter transformer with no more than 500 GPUs [42].

Be there as it may, a question emerges in our mind: is it possible to train an extreme-scale model with only a decent amount of resources, say training a 10-trillion-parameter models with 500 GPUs? Such training requires large memory for the stunning amount of parameters, including weights, gradients, and even optimizer states. Tackling the problem requires the utilization of external memory except for GPU memory, for instance CPU memory or even NVMe memory [31; 28]. These methods resolve the problem of high memory footprint, but instead their extra cost is low training efficiency caused by the frequent swap in-and-out between memories.

In this paper, we treat the training of a 10-trillion-parameter transformer model with around 500 GPUs as a case of extreme-scale training under limited resources. To this end, we propose an extremely simple two-stage training strategy called "Pseudo-to-Real" to improve training efficiency of high-memory-footprint-required transformer models for faster model convergence. This strategy is compatible with architectures built by stacking layers with an identical structure, including dense models like GPT [24; 25; 6], BERT [11], or sparse expert models like M6 [19; 42]. It first trains a relatively small model but with a computation graph of a 10-trillion-parameter model through parameter sharing across layers. Then it builds a 10-trillion-parameter model and initializes the weights of each layer with the trained weights of previous model. In this way, fast convergence can be realized in the first stage as the training costs much less memory and speeds up with large batches. Parameter sharing that addresses the communication overhead improves training speed as well. The second-stage training is responsible for the final convergence. To demonstrate the effectiveness of the training strategy, we apply it to small models to reduce time and computation costs and conduct extensive upstream and downstream experiments, including language modeling and text generation.

Furthermore, in order to make full use of CPU and GPU memory, we demonstrate a layer-wise swap-in-and-out method to increase the efficiency of CPU offloading. CPU offloading offloads data and compute to the CPU memory and it should make minimum transports of models and maximize memory savings on GPU. To the objective of effective memory footprint management, we propose a granular CPU offloading mechanism to control the granularity of transports between CPU and GPU memory.

Contributions:

- Pseudo-to-Real...
- CPU offloading...

2 Related Work

Large Scale Pretraining In recent years, pretrained language models with growing magnitudes of parameters are successively proposed, keeping to raise the validated upper limit of scaling law for model capacity w.r.t the parameter scale [13]. The earlier milestones of extreme-large models come from GPT-2 [25] and Megatron-LM [35], which demonstrates that scaling the transformer model up to billions of parameters can result in improvement on language model benchmarks [20; 22]. Turing-NLG [33], as a successor, implements a 17-billion-parameter transformer and achieves further lower perplexity. Similar phenomena are also observed on classification language tasks by T5 model [26]. The GPT-3 [6] pushes the boundary of model scale to 1750 billion parameters and demonstrates its striking effectiveness on downstream tasks in even zero-shot settings. The trend of large scale pretraining has reached to other languages [41; 47; 45] and related tasks involving other modalities like image generation [29] and code generation [7]. Along with the benefit from increasing the model scale, the concern of unaffordable pretraining cost in time, computation resource and energy keeps emerging [23; 5], resulting in the strong demand for more efficient and greener large-scale pretraining [34].

Methods to Speedup Training In previous works, the methodologies to speedup large-scale pretraining are proposed from various perspectives. To reduce the computational cost during training, some works seek to introduce in more sparsity for designing the model. Expert parallelism, proposed by Lepikhin et al. [17] and Fedus et al. [12], distributes sparsely activated feed-forward experts to multiply the parameter scale and model capacity without significantly increasing the forward

and backward FLOPs. For pretraining longer sequences, several works propose sparse transformer attention modules to efficiently compute interactions [8; 4; 44]. Aside from sparsed attention, there are also other works attempting to reduce the complexity of softmax in attention using different approximation strategies [39; 14; 15; 9].

In addition to reducing the amount of computation in a single iteration, another route to speedup training is to reduce the needed iterations for model convergence. Child et al. [8] and Xiong et al. [40] propose to put forward the layernorm operations in transformer blocks for more stabilized and faster convergence. You et al. [43] employs layerwise adaptive optimizer to enable super-large pretraining batches. Zhang & He [46] progressively increases the layer dropping rate in a stocastic manner which significantly speedups pretraining.

3 APPROACH

This section describes our methods to train an extreme-scale model. Elements of our approach includes the two-stage training strategy as well as the layer-wise offloading method.

3.1 MODEL ARCHITECTURE

Choice in model architecture depends on several factors. First, the architecture should contain a sequence of stacking layers, as the sequential structure enables cross-layer parameter sharing. Thus we prefer a simple encoder or decoder architecture instead of a relatively complex encoder-decoder framework where there are cross attention that brings extra parameters and incurs difficulties in activation checkpointing. Second, a model of such architecture should be compatible with different types of downstream tasks including understanding and generation, and it is even better that it can be compatible with multiple modalities. Third, while we mention that dense models and sparse expert models are two main tracks of large-scale pretraining, we prefer the model that can flexibly become whether dense or sparse expert models. Therefore, we select M6 [19] as an option, and we evaluate the effects of our method on M6 of different scales and types.

M6 is built with stacking layers of transformer, which includes self attention and feed-forward neural nets (FFN). For the transformation from dense models to sparse expert models, one should only replace FFNs with the Mixture-of-Expert layers (MoE). MoE consists of multiple experts, which are usualy FFNs distributed on different devices. A gating network decides the dispatching and combining behaviors of each token and thus tokens can be processed in diverse devices. Such mechanism is a combination of data parallelism and expert parallelism, and thus it is highly efficient though with large model capacity. For the training, to realize the learning of both understanding and generation, the model is trained with text denoising and language modeling on plain text data and with image-based text denoising and image captioning on multimodal data. The models is compatible with different types of downstream tasks and it can process information of multiple modalities.

3.2 PSEUDO-TO-REAL

This section demonstrates the details of "Pseudo-to-Real" two-stage training strategy that enables fast training of high-memory-footprint-required transformer models. As mentioned above, we take M6 as an example to demonstrate its effectiveness in models of different types and scales. The strategy consists of two stages. The first stage trains a model with many fewer parameters but with a large computation graph ("Pseudo Giant"), and the second stage trains a large model initialized with the weights from the previous stage ("Real Giant"). Thus we name the strategy "Pseudo-to-Real".

3.2.1 "PSEUDO" STAGE: LAYERWISE PARAMETER SHARING

The core of "Pseudo" stage is to train a Pseudo-Giant that shares parameters across layers. Cross-layer parameter sharing has proved successful in maintaining satisfactory performance while keeping a much smaller amount of parameters. The method was first mentioned in the original Transformer [37], and Dehghani et al. [10] and Bai et al. [2] both illustrated that it is effective for vanilla transformer with encoder-decoder work. Lan et al. [16] introduced the method to pretraining and proposed a lite BERT with different sharing techniques. Owing to its effectiveness, we introduce it to training an

extreme-scale model, and we hypothesize that the first-stage training can gain benefits from cross-parameter sharing as it can address communication overhead and it consumes much less memory footprint. Also, as its training is not bounded by memory, it can be trained with large batches for acceleration.

Suppose we build a M6 model with L layers that shares parameters across all layers. The Pseudo Giant though consists a computation graph of a L-layer transformer, yet its number of weight parameters and optimizer states should be $1/L$ of those of the original one. As to the gradients, we can accumulate the gradients of each layer in the backward computation process, and therefore the amount of gradients becomes $2/L$ of the original one. Such saving in memory enables much faster training with larger batches. Also, it is capable to use fewer resources even due to lower memory consumption.

Note that this rule is also available to sparse expert models. Conventional MoE models [17; 12; 42; 32] should determine the number of devices for the expert distribution in advance, and the configuration should be constant in the following stages, including finetuning and inference. While in the case of Pseudo Giant, we can distribute more experts on each device to make better use of GPU memory. In the next stage, we apply splitting on the experts so that the two stages of training can be conducted under different amounts of GPU resources. For example, we can train a Pseudo Giant with 512 experts on 256 GPU devices where each has 2 experts, and in the next stage, we can train a Real Giant on 512 GPU devices where each has 1 expert.

3.2.2 "REAL" STAGE: DELINKING THE SHARED PARAMETERS

In comparison with the Pseudo Giant with cross-layer parameter sharing, we name a large model without this technique "Real Giant". Both Pseudo Giant and Real Giant have the identical computation graph but they have different number of parameters, and thus we discover how to build a connection between the two models. Given a Pseudo Giant fully trained until convergence, we propose to apply delinking of the shared parameters to make the Real Giant benefit Pseudo Giant. Specifically, we intend to initialize the weights of Real Giant with those of Pseudo Giant. Embedding initialization can be restored directly, but the layer weights should be treated specially. In practice, there is only one layer of weights θ_{shared} in Pseudo Giant, and there are L layers of weights $\{\theta_1, \theta_2, \cdots, \theta_L\}$ in Real Giant. Thanks to their identical structure, each layer of Real Giant can be initialized with θ_{shared}. Without further training, this model is equivalent to a fully-trained Pseudo Giant.

The key difference from training a large model from scratch is that it can be trained from the converged point of Pseudo Giant, and the time for model convergence from a high level can be saved. This extremely simple training strategy is highly beneficial for the high-memory-footprint-required large models, especially extreme-scale models like the 10-trillion-parameter M6. While the first stage of training saves much time for faster convergence, we can use a decent amount of computational resources in this stage as lower efficiency in this stage becomes acceptable. Therefore, in the practice of training a 10-trillion-parameter M6, we apply CPU offloading to utilize CPU memory and thus we need fewer resources like 512 GPU devices instead of 10 times of it.

3.2.3 GRANULAR CPU OFFLOADING

Another important problem to address is that we should achieve the highest efficiency as possible while applying CPU offloading. Notably we should maximize GPU memory usage and minimizes transports between different memory regions. Therefore, we propose a granular CPU offloading mechanism to address this issue. To be more specific, we can control the granularity of the transport of models between CPU and GPU memory, say layer-wise control of the transport. In this way, the model can select which layers to offload to achieve the possible best training efficiency.

3.2.4 TIMING FOR SWITCHING

A question naturally emerges: when should we switch from the Pseudo stage to the Real stage? As mentioned above, the greatest advantage of Pseudo stage for training is the significantly faster convergence speed. Yet the performance of Pseudo Giant is bounded by its limitation in the amount of parameters. Training Pseudo Giant until its final convergence apparently incurs much waste of time.

Table 1: Model refers to the types of model. d_{model} and d_{ff} refer to the hidden size and intermediate size. L refers to the number of layers in the computation graph, and l refers to the number of transformer layers with parameters. #Heads refers to the number of heads in self attention. #Params refers to the total number of model parameters.

Model	d_{model}	d_{ff}	l/L	#Heads	#Params
Base	1024	4096	24/24	16	350M
Pseudo	1024	16384	1/36	16	1.48B
Real	1024	16384	36/36	16	1.48B

Thus we present a simple strategy for the stage switching based on the convergence speed. We train a Pseudo Giant with cross-layer parameter sharing and a Real Giant without it both from scratch. Then we compare their convergence speed on the time basis. The best timing for switching should be the moment when the convergence speed of Real Giant surpasses that of Pseudo Giant. In practice we select the checkpoint saved at the time step closest to the best timing for the next-stage training.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

We conduct experiments on the 1.4B-parameter M6 model for pretraining and finetuning to analyze model competence in upstream and downstream tasks. Following the classical data setup for pretraining and finetuning, we pretrain the model on BookCorpus [49] and English Wikipedia [11], which are corpora with around 16GB plain texts. We evaluate the upstream performance by the negative log perplexity evaluation of a held-out training set, and we evaluate its downstream performance on language modeling, text classification, and text summarization, which cover a wide range of capabilities including language modeling, understanding and generation. To be more specific, the downstream datasets include:

- **WikiText103**: a classical language modeling evaluation benchmark dataset that consists of Wikipedia articles. We evaluate the Perplexity (PPL) of the pretrained model, which is a per-token exponential cross-entropy loss that reflects probability distribution over texts.
- **Gigaword**: a dataset for summarization which consists of around 3.8M articles and summaries, and an effective benchmark to evaluate the model's capability in text generation for abstractive summarization.
- **SST-2**: a sentiment classification dataset that evaluates the model's capability in text classification. The training set consists of around 67K sample pairs of texts and labels. We use the development set for evaluation.

We validate the effectiveness of "Pseudo-to-Real" training strategy by implementing smaller models with lower computation and time costs so that we can conduct extensive analyses on both upstream and downstream performance. To satisfy the requirements of high memory footprint, we conduct experiments on a 1.4B-parameter model, the largest model trained on a single NVIDIA V100-32GB [31]. The corresponding Pseudo Giant is a model with the same computation graph but shares parameters across layers.

Following Radford et al. [24] and Lewis et al. [18], we use a vocabulary of around $50,000$ subwords. We build a model with 36 layers of transformer, whose hidden size is 1024 and intermediate size is 16384. We name the one with cross-layer parameter sharing "Pseudo" and the one without it "Real" in Table 1. We use "P2R" referring to "Pseudo-to-Real" to represent the model trained with both Pseudo and Real stage. The total number of parameters of Real Giant is around 1.4 billion and that of its corresponding Pseudo Giant is around 89 million. In comparison, the M6 baseline model has a smaller intermediate size of 4096 and consists of 24 layers. We name it "Base" in Table 1.

Following the common practice in pretraining [11; 1], we apply Adam

Experimental results are consistent with our hypothesis that Pseudo-to-Real training can speed up training effectively and the model trained with both stages can match the performance of the Real

Giant trained from scratch in both upstream and downstream tasks. We add other strong baseline pretrained models for the three tasks for comparison. While M6-base model matches the performance of pretrianed models, including BERT, UniLM, Megatron-LM, etc., the Real Giant with larger model scale achieve better performance in all tasks, reflecting the availability of neural scaling law. Most importantly, we find that P2R can achieve similar or slightly undercompete the Real Giant in downstream tasks, and they have highly similar performance in the upstream PPL evaluation. As to the training speed, in comparison with Real Giant, the training speed of P2R is significantly faster. To match the training efficiency of P2R, Real Giant requires at least 5 times of GPU resources, incurring substantially higher financial and environmental costs.

5 Training a 10-Trillion-Parameter Model

6 Conclusion and Future Work

References

[1] Yinhan Liu an. Roberta: A robustly optimized BERT pretraining approach. ArXiv preprint, abs/1907.11692, 2019.

[2] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. Deep equilibrium models. In Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, pp. 688–699, 2019.

[3] Mandeep Baines, Shruti Bhosale, Vittorio Caggiano, Naman Goyal, Siddharth Goyal, Myle Ott, Benjamin Lefaudeux, Vitaliy Liptchinsky, Mike Rabbat, Sam Sheiffer, Anjali Sridhar, and Min Xu. Fairscale: A general purpose modular pytorch library for high performance and large scale training. `https://github.com/facebookresearch/fairscale`, 2021.

[4] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. arXiv preprint arXiv:2004.05150, 2020.

[5] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258, 2021.

[6] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual, 2020.

[7] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde, Jared Kaplan, Harri Edwards, Yura Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. arXiv preprint arXiv:2107.03374, 2021.

[8] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. arXiv preprint arXiv:1904.10509, 2019.

[9] Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamás Sarlós, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J. Colwell, and Adrian Weller. Rethinking attention with performers. In 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net, 2021.

[10] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. In 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net, 2019.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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), pp. 4171–4186, Minneapolis, Minnesota, 2019. Association for Computational Linguistics.

[12] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. ArXiv preprint, abs/2101.03961, 2021.

[13] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. arXiv preprint arXiv:2001.08361, 2020.

[14] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event, volume 119 of Proceedings of Machine Learning Research, pp. 5156–5165. PMLR, 2020.

[15] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net, 2020.

[16] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. In 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net, 2020.

[17] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. ArXiv preprint, abs/2006.16668, 2020.

[18] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 7871–7880, Online, 2020. Association for Computational Linguistics.

[19] Junyang Lin, Rui Men, An Yang, Chang Zhou, Ming Ding, Yichang Zhang, Peng Wang, Ang Wang, Le Jiang, Xianyan Jia, et al. M6: A chinese multimodal pretrainer. arXiv preprint arXiv:2103.00823, 2021.

[20] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net, 2017.

[21] Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Anand Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, et al. Efficient large-scale language model training on gpu clusters. arXiv preprint arXiv:2104.04473, 2021.

[22] Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1525–1534, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1144.

[23] David A. Patterson, Joseph Gonzalez, Quoc V. Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David R. So, Maud Texier, and Jeff Dean. Carbon emissions and large neural network training. ArXiv preprint, abs/2104.10350, 2021.

[24] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. URL https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf, 2018.

[25] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, 2019.

[26] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research, 21:1–67, 2020.

[27] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In SC20: International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1–16. IEEE, 2020.

[28] Samyam Rajbhandari, Olatunji Ruwase, Jeff Rasley, Shaden Smith, and Yuxiong He. Zero-infinity: Breaking the gpu memory wall for extreme scale deep learning. arXiv preprint arXiv:2104.07857, 2021.

[29] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. arXiv preprint arXiv:2102.12092, 2021.

[30] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 3505–3506, 2020.

[31] Jie Ren, Samyam Rajbhandari, Reza Yazdani Aminabadi, Olatunji Ruwase, Shuangyan Yang, Minjia Zhang, Dong Li, and Yuxiong He. Zero-offload: Democratizing billion-scale model training. arXiv preprint arXiv:2101.06840, 2021.

[32] Stephen Roller, Sainbayar Sukhbaatar, Arthur Szlam, and Jason Weston. Hash layers for large sparse models. arXiv preprint arXiv:2106.04426, 2021.

[33] Corby Rosset. Turing-nlg: A 17-billion parameter language model by microsoft. Microsoft Research Blog, 2020.

[34] Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. Green AI. Commun. ACM, 63 (12):54–63, 2020.

[35] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. arXiv preprint arXiv:1909.08053, 2019.

[36] Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, et al. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. arXiv preprint arXiv:2107.02137, 2021.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pp. 5998–6008, 2017.

[38] Ang Wang, Xianyan Jia, Le Jiang, Jie Zhang, Yong Li, and Wei Lin. Whale: A unified distributed training framework. ArXiv preprint, abs/2011.09208, 2020.

[39] Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. arXiv preprint arXiv:2006.04768, 2020.

[40] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture. In Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event, volume 119 of Proceedings of Machine Learning Research, pp. 10524–10533. PMLR, 2020.

[41] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mT5: A massively multilingual pre-trained text-to-text transformer. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 483–498, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.41.

[42] An Yang, Junyang Lin, Rui Men, Chang Zhou, Le Jiang, Xianyan Jia, Ang Wang, Jie Zhang, Jiamang Wang, Yong Li, et al. M6-t: Exploring sparse expert models and beyond. arXiv preprint arXiv:2105.15082, 2021.

[43] Yang You, Jing Li, Sashank J. Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training BERT in 76 minutes. In 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net, 2020.

[44] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual, 2020.

[45] Wei Zeng, Xiaozhe Ren, Teng Su, Hui Wang, Yi Liao, Zhiwei Wang, Xin Jiang, ZhenZhang Yang, Kaisheng Wang, Xiaoda Zhang, et al. Pangu-α: Large-scale autoregressive pretrained chinese language models with auto-parallel computation. arXiv preprint arXiv:2104.12369, 2021.

[46] Minjia Zhang and Yuxiong He. Accelerating training of transformer-based language models with progressive layer dropping. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual, 2020.

[47] Zhengyan Zhang, Yuxian Gu, Xu Han, Shengqi Chen, Chaojun Xiao, Zhenbo Sun, Yuan Yao, Fanchao Qi, Jian Guan, Pei Ke, et al. Cpm-2: Large-scale cost-effective pre-trained language models. arXiv preprint arXiv:2106.10715, 2021.

[48] Zhengyan Zhang, Xu Han, Hao Zhou, Pei Ke, Yuxian Gu, Deming Ye, Yujia Qin, Yusheng Su, Haozhe Ji, Jian Guan, et al. Cpm: A large-scale generative chinese pre-trained language model. AI Open, 2:93–99, 2021.

[49] Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In 2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015, pp. 19–27. IEEE Computer Society, 2015.

A APPENDIX

You may include other additional sections here.