Supplement for the paper
AI Computing Systems for Large Language Models Training

Zhen-Xing Zhang[1,2] (张振兴), Yuan-Bo Wen[2] (文渊博), Han-Qi Lyu[1,2,3] (吕涵祺),
Chang Liu[3] (刘畅), Rui Zhang[2] (张蕊), Xia-Qing Li[2] (李夏青),
Chao Wang[1] (王超), *Senior Member, CCF*, Zi-Dong Du[2,4] (杜子东), *Senior Member, CCF*,
Qi Guo[2] (郭崎), *Senior Member, CCF*, Ling Li[5] (李玲), *Senior Member, CCF*,
Xue-Hai Zhou[1] (周学海), *Senior Member, CCF*, Yun-Ji Chen[2,6,*] (陈云霁), *Fellow, CCF*

[1] *School of Computer Science and Technology, University of Science and Technology of China, Hefei 230026, China*

[2] *State Key Lab of Processors, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China*

[3] *Cambricon Technologies. Beijing 100191, China*

[4] *Shanghai Innovation Center for Processor Technologies, Shanghai 201210, China*

[5] *Intelligent Software Research Center, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China*

[6] *University of Chinese Academy of Sciences, Beijing 101408, China*

We have selectively added some related references in this supplement for those that could not be included in the published paper.

About Section 2

Models:

- ChatGPT [1]
- Llama3 [2, 3]
- GPT-4o [4]
- Sora [5]

Normalization methods:

- LayerNorm [6]
- RMSNorm [7]
- DeepNorm [8]

Position embedding methods:

- absolute position embedding [9]
- relative position embedding [10, 11]
- Attention with Linear Biases (ALiBi) [12]
- Rotary Position Embedding (RoPE) [13]

Activation functions:

- ReLU [14]
- GeLU [15]
- Swish [16]
- ReGLU [17]
- GEGLU [17]
- SwiGLU [17]

Learning rate adjustment:

- warmup [18]
- LARS [19]
- LAMB [20]
- cosine delay [21]

Optimizers:

- Adam [22]
- AdamW [23]
- AdaFactor [24]
- SGD

About Section 3

The data types used in matrix operations

- FP16
- BF16 [58]
- TF32 [59]
- FP8 [60]
- FP6 and FP4 [61]

Accelerators

- AMD MI300 GPU [62]
- Cambricon MLU 370 [63]
- Google TPU v4 [64]
- Intel Gaudi 2 AI accelerator [65]
- NVIDIA V100 GPU and DGX-1 server [66]
- NVIDIA A100 GPU and DGX-A100 server [67]
- NVIDIA H100 GPU and DGX-H100 server [68]

Network topology

- Fat-Tree [69]
- BCube [70]
- DragonFly [71]

About Section 4

As shown in the following Table, we have added supplementary reference materials to Table 5 in Section 4.1 of the original paper, including web links and reference papers.

Table 1. LLM Training Programming Frameworks (Table 5 in Section 4.1 of the original paper)

Programming Framework	Basic Source Components	Features	Parallel Mode
Mesh-TensorFlow [25,26]	TensorFlow	Distributed tensor computation	MP
Torchgpipe [27,28]	PyTorch	Micro-batch pipeline by Gpipe	PP
Fairscale [29,30] (Meta)	PyTorch	Fully Sharded Data Parallel (FSDP)	DP
Deepspeed [31–34] (Microsoft)	PyTorch	Zero Redundancy Optimizer (ZeRO), ZeRO series	DP, MP, PP
Megatron-LM [35–37] (NVIDIA)	PyTorch	Interleaved 1F1B pipeline, partitioned transformer	DP, MP, PP
Megatron-DeepSpeed [38] (Microsoft)	Forked from Megatron-LM	DeepSpeed integrated in Megatron	DP, MP, PP
Megatron-LLaMA [39] (Alibaba)	Forked from Megatron-LM	LLaMA, improved sharding method, distributed checkpoint	DP, MP, PP
veGiantModel [40] (ByteDance)	Megatron-LM and DeepSpeed	Improved communication, customized pipline partitioning	DP, MP, PP
PaddleFleetX [41] (Baidu)	PaddlePaddle	Hybrid parallel (4D), unified trainer	DP, MP, PP
OneFlow [42,43] (OneFlow)	OneFlow	Split, broadcast and partial-value (SBP)	DP, MP, PP
EPL [44] or Whale [45] (Alibaba)	TensorFlow	Easy parallel annotation	DP, MP, PP
AxoNN [46,47] (Axonn-AI)	PyTorch	Asynchrony and message-driven execution, CPU memory offloading	DP, PP
Varuna [48,49] (Microsoft)	PyTorch	Elastic training, low-bandwidth network, job morphing	DP, PP
Colossal-AI [50,51] (HPC-AI)	PyTorch	Multi parallelism strategies, memory offloading (PatrickStar)	DP, MP, PP
Merak [52,53]	PyTorch	Automatic model partitioner	DP, MP, PP
Bamboo [54,55]	DeepSpeed	Elastic training, redundant computation, preemptible instance	DP, PP
Oobleck [56,57] (UMich)	PyTorch, DeepSpeed, Merak	Elastic training, pipeline template, dynamic reconfiguration	DP, MP, PP

References

[1] OpenAI. Chatgpt. https://openai.com/blog/chatgpt, 2022.

[2] Meta llama 3. https://github.com/meta-llama/llama3.

[3] Meta llama 3.1. https://ai.meta.com/blog/meta-llama-3-1/.

[4] Openai sora. https://openai.com/index/hello-gpt-4o/.

[5] Openai sora. https://openai.com/index/video-generation-models-as-world-simulators/.

[6] Ba J L, Kiros J R, Hinton G E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[7] Zhang B, Sennrich R. Root mean square layer normalization. *Advances in Neural Information Processing Systems (NIPS)*, 2019, 32.

[8] Wang H, Ma S, Dong L, Huang S, Zhang D, Wei F. Deepnet: Scaling transformers to 1,000 layers. *arXiv preprint arXiv:2203.00555*, 2022.

[9] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser L, Polosukhin I. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[10] Shaw P, Uszkoreit J, Vaswani A. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018.

[11] Huang Z, Liang D, Xu P, Xiang B. Improve transformer models with better relative position embeddings. *arXiv preprint arXiv:2009.13658*, 2020.

[12] Press O, Smith N A, Lewis M. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*, 2021.

[13] Su J, Lu Y, Pan S, Murtadha A, Wen B, Liu Y. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

[14] Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011, pp. 315–323.

[15] Hendrycks D, Gimpel K. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[16] Ramachandran P, Zoph B, Le Q V. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

[17] Shazeer N. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

[18] Goyal P, Dollár P, Girshick R, Noordhuis P, Wesolowski L, Kyrola A, Tulloch A, Jia Y, He K. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[19] You Y, Gitman I, Ginsburg B. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

[20] You Y, Li J, Reddi S, Hseu J, Kumar S, Bhojanapalli S, Song X, Demmel J, Keutzer K, Hsieh C J. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019.

[21] He T, Zhang Z, Zhang H, Zhang Z, Xie J, Li M. Bag of tricks for image classification with convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2019, pp. 558–567.

[22] Kingma D P, Ba J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[23] Loshchilov I, Hutter F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[24] Shazeer N, Stern M. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018, pp. 4596–4604.

[25] Mesh-tensorflow. https://github.com/tensorflow/mesh, 2019.

[26] Shazeer N, Cheng Y, Parmar N, Tran D, Vaswani A, Koanantakool P, Hawkins P, Lee H, Hong M, Young C et al. Mesh-tensorflow: Deep learning for supercomputers. *Advances in Neural Information Processing Systems (NIPS)*, 2018, 31.

[27] Torchgpipe. https://github.com/kakaobrain/torchgpipe, 2019.

[28] Kim C, Lee H, Jeong M, Baek W, Yoon B, Kim I, Lim S, Kim S. torchgpipe: On-the-fly pipeline parallelism for training giant models. *arXiv preprint arXiv:2004.09910*, 2020.

[29] Fairscale: A general purpose modular pytorch library for high performance and large scale training. https://github.com/facebookresearch/fairscale, 2021.

[30] Introducing pytorch fully sharded data parallel (fsdp) api. https://pytorch.org/blog/introducing-pytorch-fully-sharded-data-parallel-api/, 2022.

[31] Rasley J, Rajbhandari S, Ruwase O, He Y. 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 and Data Mining (SIGKDD)*, 2020, pp. 3505–3506.

[32] Li C, Yao Z, Wu X, Zhang M, Holmes C, Li C, He Y. Deepspeed data efficiency: Improving deep learning model quality and training efficiency via efficient data sampling and routing. *arXiv preprint arXiv:2212.03597*, 2022.

[33] Aminabadi R Y, Rajbhandari S, Awan A A, Li C, Li D, Zheng E, Ruwase O, Smith S, Zhang M, Rasley J et al. Deepspeed-inference: enabling efficient inference of transformer models at unprecedented scale. In *International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, 2022, pp. 1–15.

[34] Rajbhandari S, Li C, Yao Z, Zhang M, Aminabadi R Y, Awan A A, Rasley J, He Y. Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation ai scale. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2022, pp. 18332–18346.

[35] Megatron-lm. https://github.com/NVIDIA/Megatron-LM, 2019.

[36] Shoeybi M, Patwary M, Puri R, LeGresley P, Casper J, Catanzaro B. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

[37] Narayanan D, Shoeybi M, Casper J, LeGresley P, Patwary M, Korthikanti V, Vainbrand D, Kashinkunti P, Bernauer J, Catanzaro B et al. Efficient large-scale language model training on gpu clusters using megatron-lm. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, 2021, pp. 1–15.

[38] Megatron-deepspeed. https://github.com/microsoft/Megatron-DeepSpeed, 2019.

[39] Megatron-llama. https://github.com/alibaba/Megatron-LLaMA, 2022.

[40] vegiantmodel. https://github.com/volcengine/veGiantModel, 2021.

[41] Paddlefleetx: An easy-to-use and high-performance one-stop tool for deep learning. https://github.com/PaddlePaddle/PaddleFleetX, 2022.

[42] Oneflow. https://github.com/OneFlow-Inc/oneflow, 2021.

[43] Yuan J, Li X, Cheng C, Liu J, Guo R, Cai S, Yao C, Yang F, Yi X, Wu C et al. Oneflow: Redesign the distributed deep learning framework from scratch. *arXiv preprint arXiv:2110.15032*, 2021.

[44] Easy parallel library (epl). https://github.com/alibaba/EasyParallelLibrary, 2022.

[45] Jia X, Jiang L, Wang A, Xiao W, Shi Z, Zhang J, Li X, Chen L, Li Y, Zheng Z et al. Whale: Efficient giant model training over heterogeneous gpus. In *2022 USENIX Annual Technical Conference (ATC)*, 2022, pp. 673–688.

[46] Axonn. https://github.com/axonn-ai/axonn, 2022.

[47] Singh S, Bhatele A. Axonn: An asynchronous, message-driven parallel framework for extreme-scale deep learning. In *2022 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2022, pp. 606–616.

[48] Varuna. https://github.com/microsoft/varuna, 2022.

[49] Athlur S, Saran N, Sivathanu M, Ramjee R, Kwatra N. Varuna: scalable, low-cost training of massive deep learning models. In *Proceedings of the 17th European Conference on Computer Systems (EuroSys)*, 2022, pp. 472–487.

[50] Colossal-ai. https://github.com/hpcaitech/ColossalAI, 2023.

[51] Li S, Liu H, Bian Z, Fang J, Huang H, Liu Y, Wang B, You Y. Colossal-ai: A unified deep learning system for large-scale parallel training. In *Proceedings of the 52nd International Conference on Parallel Processing (ICPP)*, 2023, pp. 766–775.

[52] Merak. https://github.com/HPDL-Group/Merak, 2023.

[53] Lai Z, Li S, Tang X, Ge K, Liu W, Duan Y, Qiao L, Li D. Merak: An efficient distributed dnn training framework with automated 3d parallelism for giant foundation models. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 2023, 34(5):1466–1478.

[54] Bamboo. https://github.com/uclasystem/bamboo, 2023.

[55] Thorpe J, Zhao P, Eyolfson J, Qiao Y, Jia Z, Zhang M, Netravali R, Xu G H. Bamboo: Making preemptible instances resilient for affordable training of large dnns. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2023, pp. 497–513.

[56] Oobleck. https://github.com/SymbioticLab/Oobleck, 2023.

[57] Jang I, Yang Z, Zhang Z, Jin X, Chowdhury M. Oobleck: Resilient distributed training of large models using pipeline templates. In *Proceedings of the 29th Symposium on Operating Systems Principles (SOSP)*, 2023, pp. 382–395.

[58] Bfloat16: The secret to high performance on cloud tpus. https://cloud.google.com/blog/products/ai-machine-learning/bfloat16-the-secret-to-high-performance-on-cloud-tpus, 2019.

[59] Accelerating ai training with nvidia tf32 tensor cores. https://developer.nvidia.com/blog/accelerating-ai-training-with-tf32-tensor-cores/, 2021.

[60] Using fp8 with transformer engine. https://docs.nvidia.com/deeplearning/transformer-engine/user-guide/examples/fp8_primer.html, 2022.

[61] Nvidia blackwell architecture. https://resources.nvidia.com/en-us-blackwell-architecture.

[62] AMD. Amd mi300 series accelerators. https://www.amd.com/en/products/accelerators/instinct/mi300.html, 2023.

[63] Cambricon. Cambricon mlu 370. https://www.cambricon.com/docs/product_docs/mlu370_x8/1.0.0/index.html, 2022.

[64] Jouppi N, Kurian G, Li S, Ma P, Nagarajan R, Nai L, Patil N, Subramanian S, Swing A, Towles B, Young C, Zhou X, Zhou Z, Patterson D A. Tpu v4: An optically reconfigurable supercomputer for machine learning with hardware support for embeddings. In *Proceedings of the 50th Annual International Symposium on Computer Architecture (ISCA)*, 2023, pp. 1–14.

[65] Intel. Intel gaudi 2 ai accelerators whitepaper. https://habana.ai/wp-content/uploads/2023/10/Intel-Gaudi2-AI-Accelerators-whitepaper.pdf, 2023.

[66] NVIDIA. Nvidia tesla v100 gpu architecture. https://images.nvidia.com/content/volta-architecture/pdf/volta-architecture-whitepaper.pdf, 2020.

[67] NVIDIA. Nvidia a100 tensor core gpu architecture. https://images.nvidia.com/aem-dam/en-zz/Solutions/data-center/nvidia-ampere-architecture-whitepaper.pdf, 2020.

[68] NVIDIA. Nvidia h100 tensor core gpu architecture. https://resources.nvidia.com/en-us-tensor-core/gtc22-whitepaper-hopper, 2022.

[69] Al-Fares M, Loukissas A, Vahdat A. A scalable, commodity data center network architecture. *ACM SIGCOMM computer communication review*, 2008, 38(4):63–74.

[70] Guo C, Lu G, Li D, Wu H, Zhang X, Shi Y, Tian C, Zhang Y, Lu S. Bcube: a high performance, server-centric network architecture for modular data centers. In *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, 2009, pp. 63–74.

[71] Kim J, Dally W J, Scott S, Abts D. Technology-driven, highly-scalable dragonfly topology. *ACM SIGARCH Computer Architecture News*, 2008, 36(3):77–88.