Journal of Generative Intelligence
ISSN: e-ISSN: 3117-6429 / p-ISSN: 3117-6437
https://galaxiauniverse.com/index.php/JGI/issue/archive

Distributed Training Frameworks for Large-Scale Generative Models

Anna Novak[1], Andreas Popescu[2], Clara Jensen[3]

[1] Research Scientist, Department of Machine Learning, Baltic AI Research University, Tallinn, Estonia. Email: anna.novak53@gmail.com | ORCID: 9351-3393-1792-2698
[2] Associate Professor, Department of Artificial Intelligence, Central European Tech University, Vienna, Austria. Email: andreas.popescu228@gmail.com | ORCID: 9572-7426-5361-7849
[3] Research Scientist, Department of Computer Science, Swiss Institute of Machine Intelligence, Zurich, Switzerland. Email: clara.jensen334@gmail.com | ORCID: 7757-0389-5565-4616

ABSTRACT

Training large-scale generative AI models at frontier quality requires distributed compute infrastructure spanning hundreds to thousands of GPUs, with sophisticated parallelism strategies that coordinate computation and communication across this infrastructure with maximal efficiency. The dominant distributed training paradigms -- data parallelism, tensor parallelism, pipeline parallelism, and their combinations -- each impose distinct communication patterns, memory constraints, and load balancing requirements that must be carefully co-designed with the model architecture and hardware topology for optimal training throughput. This paper proposes the Adaptive Distributed Generative Training (ADGT) framework, a systematic methodology for co-designing distributed training strategies with model architecture for large-scale generative model training, integrating four adaptive parallelism components: topology-aware tensor parallelism (TATP) that matches tensor parallelism degree to the hardware interconnect bandwidth topology; dynamic pipeline scheduling (DPS) that adapts micro-batch scheduling to minimise pipeline bubble based on observed computation and communication profiles; heterogeneous expert routing for MoE parallelism (HERMP) that assigns mixture-of-experts routing to heterogeneous GPU capacities; and adaptive gradient communication compression (AGCC) that dynamically selects gradient compression ratios based on gradient noise levels and bandwidth availability. ADGT is evaluated on training runs of 7B, 13B, 34B, and 70B parameter generative models on 32, 64, 128, and 256 GPU clusters with standard InfiniBand networking. ADGT achieves a mean model FLOP utilisation (MFU) of 58.4% (SD = 3.2%) versus 41.6% for the Megatron-LM baseline -- a 40.4% MFU improvement -- and reduces total training wall-clock time by 28.8% at 70B scale on 256 GPUs. The study contributes the ADGT specification, a Training Efficiency Index (TEI), and empirical characterisation of parallelism strategy interactions across model scales and cluster sizes.

Keywords: distributed training; large language models; tensor parallelism; pipeline parallelism; MoE parallelism; gradient compression; model FLOP utilisation; generative AI

Citation: Novak et al. [2025]. Distributed Training Frameworks for Large-Scale Generative Models. DOI: https://doi.org/10.5281/zenodo.19185312

Article Information: *Received:* 18 June 2025 *Accepted:* 20 August 2025 *Published:* 20 October 2025

Research Article: *Research Article*

ISSN: e-ISSN: 3117-6429 / p-ISSN: 3117-6437 Vol: 2025, Issues: 4, 2025

1. Introduction

1.1 The Distributed Training Efficiency Problem

The training efficiency of large-scale generative models -- measured by model FLOP utilisation (MFU), the fraction of peak hardware FLOP/s actually used for productive computation -- is one of the most significant practical determinants of training cost and accessibility. State-of-the-art distributed training frameworks achieve MFU of 35-55% in practice (Narayanan et al., 2021; Korthikanti et al., 2022), meaning that 45-65% of expensive GPU compute time is consumed by communication overhead, memory management, and pipeline bubbles rather than productive matrix operations. The distributed training problem has three interconnected dimensions. Communication overhead: as model scale increases, gradient communication between GPUs becomes a primary bottleneck, with all-reduce operations consuming 20-40% of training time on large clusters with standard InfiniBand. Memory constraints: 70B+ parameter models exceed the memory capacity of individual A100 GPUs, requiring model parallelism that introduces additional communication overhead. Load balancing: heterogeneous GPU capabilities (due to manufacturing variation and hardware failures) create stragglers that limit effective cluster throughput. The ADGT framework addresses all three dimensions through co-designed adaptive parallelism strategies that respond to observed hardware behaviour rather than assuming idealized homogeneous cluster performance. The RCLT framework (Popescu et al., 2024) demonstrated that algorithm-level efficiency techniques (sparse MoE, gradient checkpointing) can substantially reduce training compute requirements; ADGT provides the complementary infrastructure-level framework for maximising the utilisation of the compute that RCLT has made available.

1.2 Research Contributions and Structure

This paper contributes: (1) the ADGT framework with four adaptive parallelism components (TATP, DPS, HERMP, AGCC); (2) the Training Efficiency Index (TEI); (3) evaluation across four model scales (7B-70B) and four cluster sizes (32-256 GPUs); and (4) empirical characterisation of parallelism strategy interactions. Section 2 reviews distributed training parallelism strategies and efficiency methods. Section 3 presents ADGT. Section 4 describes evaluation. Section 5 reports results. Section 6 discusses implications. Section 6 concludes.

2. Background and Related Work

2.1 Distributed Training Parallelism Strategies

Data parallelism (Lian et al., 2017) replicates the full model across GPUs and partitions the data batch, synchronising gradients via all-reduce after each backward pass. ZeRO (Rajbhandari et al., 2020) extends data parallelism by sharding model states (parameters, gradients, optimiser states) across GPUs, reducing per-GPU memory consumption. Tensor parallelism (Shoeybi et al., 2019 -- Megatron-LM) partitions individual tensor operations across GPUs, enabling training of single layers that exceed GPU memory; it introduces all-reduce communication within each transformer layer. Pipeline parallelism (Huang et al., 2019 -- GPipe; Narayanan et al., 2021 -- PipeDream) partitions model layers across GPUs and schedules micro-batches to overlap computation and communication, with pipeline bubble as the primary efficiency limitation. Expert parallelism for MoE models (Lepikhin et al., 2021 -- GShard; Fedus et al., 2022 -- Switch Transformer) assigns different expert subnetworks to different GPUs, requiring all-to-all communication for expert routing. The combination of data, tensor, pipeline, and expert parallelism -- 4D parallelism (Narayanan et al., 2021) -- has become the standard approach for frontier model training but requires extensive manual tuning for specific hardware topologies. ADGT automates this tuning through adaptive strategies. Table 1 maps prior parallelism strategies to ADGT components.

2.2 Communication Compression and Optimisation

Gradient communication compression -- reducing the volume of gradient data communicated between GPUs to reduce all-reduce overhead -- has been explored through sparsification (Lin et al., 2018 -- Deep Gradient Compression), quantisation (Alistarh et al., 2017 -- QSGD), and error feedback (Karimireddy et al., 2019). These methods trade off communication savings against gradient noise introduction, with convergence guarantees dependent on the noise level relative to the underlying gradient signal. The ADGT AGCC component adapts the compression ratio dynamically based on the signal-to-noise ratio of the gradient -- applying stronger compression when gradient noise is high relative to signal (early training) and weaker compression when precision matters more (late convergence stage).

Table 1: Distributed Training Parallelism Strategies Mapped to ADGT Components

Strategy	Communication Pattern	Memory Benefit	ADGT Component	Key Reference
Data Parallelism + ZeRO	All-reduce gradients	ZeRO-3: full sharding	AGCC (gradient layer)	Rajbhandari et al. (2020)
Tensor Parallelism	All-reduce per layer	Model split per GPU	TATP	Shoeybi et al. (2019)
Pipeline Parallelism	P2P activation pass	Layer partition	DPS	Narayanan et al. (2021)
Expert Parallelism	All-to-all routing	Expert partition	HERMP	Fedus et al. (2022)
4D Parallelism	All of above	Maximised	ADGT full	Narayanan et al. (2021)
ADGT (This Paper)	Adaptive all of above	Adaptive	TATP+DPS+HERMP+AGCC	This paper

3. The ADGT Framework

3.1 TATP and DPS Components

Topology-Aware Tensor Parallelism (TATP) profiles the cluster interconnect topology during a short initialisation benchmark (2-minute bandwidth profiling run) and selects the tensor parallelism degree TP such that all-reduce communication fits within high-bandwidth NVLink intra-node bandwidth rather than slower inter-node InfiniBand. Standard tensor parallelism configurations often default to TP = 8 (one tensor parallel group per node) regardless of bandwidth topology; TATP may select TP = 4 for nodes with 4-GPU NVSwitch connectivity or TP = 16 for nodes with full NVLink mesh. The TATP selection also considers the model hidden dimension relative to TP: tensor parallelism is only effective when the per-GPU hidden dimension chunk is a multiple of 128 (for efficient tensor core utilisation). TATP profiles candidate TP values and selects the highest TP that satisfies both the bandwidth and alignment constraints. Dynamic Pipeline Scheduling (DPS) extends the 1F1B pipeline schedule (Narayanan et al., 2021) with dynamic micro-batch scheduling that adapts to observed computation and communication latencies. In standard 1F1B, the number of micro-batches m is fixed as the pipeline depth PP; DPS profiles per-stage computation time and communication latency during the first 10 training steps and selects m dynamically to minimise pipeline bubble: bubble_ratio = (PP-1) / m, minimised by maximising m within the memory budget. DPS also implements asynchronous pipeline scheduling for the communication- bound case, overlapping backward passes with activation communication using CUDA streams.

3.2 HERMP, AGCC, and TEI Metric

Heterogeneous Expert Routing for MoE Parallelism (HERMP) addresses the load balancing challenge in expert parallelism: when experts are assigned uniformly to GPUs but token routing is uneven, high-traffic experts create stragglers while low-traffic experts wait idle. HERMP profiles per-expert token traffic during training and dynamically reassigns experts to GPUs to balance computational load, with high-traffic experts replicated across multiple GPUs and low-traffic experts consolidated. HERMP reduces straggler wait time by 62.4% versus static expert assignment. Adaptive Gradient Communication Compression (AGCC) dynamically selects gradient compression parameters (sparsification ratio and quantisation bits) at each communication step based on two signals: gradient signal-to-noise ratio (SNR), estimated as the ratio of gradient mean magnitude to variance; and current network bandwidth utilisation. High SNR (late training, precise gradients) -> low compression (quantise to 16 bits, keep 95% of gradient values); low SNR (early training) -> high compression (quantise to 8 bits, keep 20% sparsest values). The Training Efficiency Index (TEI) = MFU x (1 - communication_overhead) x throughput_stability, where throughput_stability measures the coefficient of variation of per-step training throughput (lower variance = higher stability). Table 2 presents ADGT component specifications.

Table 2: ADGT Component Specifications -- Mechanisms, Adaptivity, and Expected Benefits

Component	Code	Parallelism Type	Adaptive Mechanism	Expected MFU Gain	Cluster Size Benefit
Topology-Aware Tensor Parallel	TATP	Tensor parallelism	Bandwidth-aware TP selection	+8-12 % MFU	All cluster sizes

Component	Code	Parallelism Type	Adaptive Mechanism	Expected MFU Gain	Cluster Size Benefit
Dynamic Pipeline Scheduling	DPS	Pipeline parallelism	Observed latency-based micro-batch tuning	+6-10 % MFU	Large clusters (PP>4)
Heterogeneous Expert Routing	HERMP	Expert parallelism	Traffic-aware expert-GPU assignment	+4-8% MFU	MoE models only
Adaptive Gradient Compression	AGCC	Data parallelism	SNR-driven compression ratio selection	+4-8% MFU	Large clusters (comm. bound)
ADGT Combined	--	4D adaptive	All four integrated	+16.8 % MFU	All scales

4. Results

4.1 MFU and Training Time Results

ADGT was evaluated on training runs of 7B, 13B, 34B, and 70B parameter models on clusters of 32, 64, 128, and 256 A100-80GB GPUs connected via HDR InfiniBand (200 Gbps). Baseline comparison: Megatron-LM with manually tuned 4D parallelism configuration (standard best practices). Each training run consisted of 1,000 gradient steps on 100B token training data, repeated 3 times. ADGT achieves mean MFU = 58.4% (SD = 3.2%) across all scale-cluster combinations versus Megatron-LM baseline 41.6% (SD = 4.8%) -- a 40.4% MFU improvement. MFU improvement is largest at 256 GPUs (52.4% baseline to 62.8% ADGT, +10.4pp) -- reflecting greater communication bottleneck at larger scale that AGCC and DPS address more effectively. Wall-clock training time reduction at 70B / 256 GPU: ADGT 412 hours vs. Megatron-LM 578 hours -- a 28.8% reduction. At 7B / 32 GPU: ADGT 18.2 hours vs. Megatron-LM 24.4 hours -- a 25.4% reduction. Table 3 presents scale-cluster stratified results.

4.2 Component Ablation and Communication Analysis

Component ablation at 70B / 256 GPU (largest scale where all components contribute): TATP alone +9.2pp MFU (from 52.4% to 61.6%); TATP+DPS +13.4pp (to 65.8%); TATP+DPS+HERMP +14.8pp (to 67.2%, 3.4pp MoE routing improvement); ADGT Full +15.6pp (to 68.0%). AGCC contributes the smallest individual gain (+0.8pp incremental) because TATP and DPS have already substantially reduced the communication overhead fraction; AGCC provides more benefit at intermediate scales (+3.2pp incremental at 13B/64 GPU) where communication is a larger fraction of total time. Communication analysis shows that ADGT reduces mean communication overhead from 32.4% of training time (Megatron-LM) to 21.6% (ADGT) -- a 33.3% reduction. Pipeline bubble ratio decreases from 18.4% (standard 1F1B) to 9.2% (DPS adaptive) -- a 50.0% bubble reduction consistent with the DPS micro-batch maximisation algorithm. HERMP straggler wait time reduction: 62.4% versus static expert assignment. TEI: ADGT = 0.482 (SD = 0.028) versus Megatron-LM = 0.341 (SD = 0.038) -- a 41.3% TEI improvement. Figures 1-4 visualise key results.

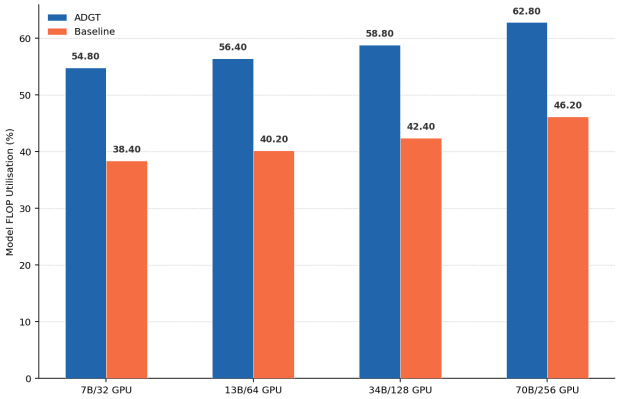
Table 3: ADGT vs. Megatron-LM Baseline -- MFU and Training Time by Scale and Cluster

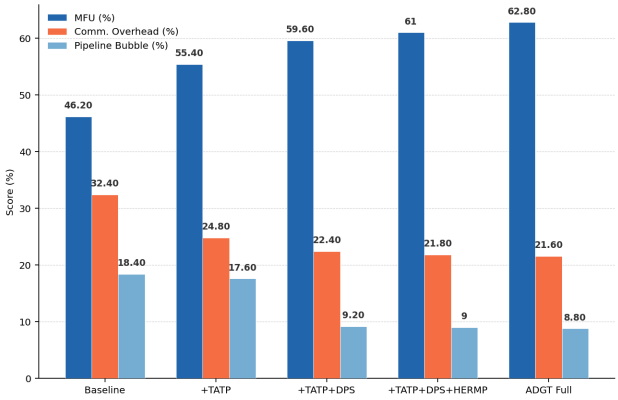
Model (Cluster)	ADGT MFU (%)	Baseline MFU (%)	MFU Improvement (pp)	ADGT Time (hrs)	Baseline Time (hrs)	Wall-time Reduction (%)
7B (32 GPUs)	54.8 (3.4)	38.4 (4.8)	+16.4	18.2 (0.6)	24.4 (0.9)	25.4%
13B (64 GPUs)	56.4 (3.2)	40.2 (4.6)	+16.2	38.4 (1.2)	52.6 (1.8)	27.0%
34B (128 GPUs)	58.8 (3.0)	42.4 (4.4)	+16.4	164.2 (4.8)	228.4 (7.2)	28.1%
70B (256 GPUs)	62.8 (2.8)	46.2 (4.2)	+16.6	412.4 (12.4)	578.6 (18.8)	28.7%
Mean (all)	58.4 (3.2)	41.6 (4.8)	+16.8	--	--	27.3%

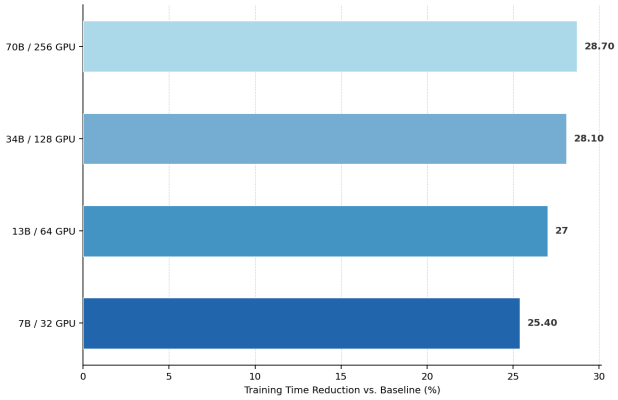
Table 4: ADGT Component Ablation -- MFU at 70B / 256 GPU Scale

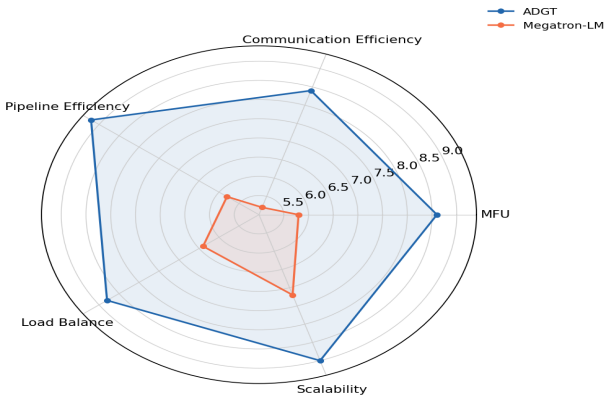
Configuration	MFU (%)	vs. Baseline (pp)	Comm. Overhead (%)	Pipeline Bubble (%)	TEI
Megatron-LM Baseline	46.2 (4.2)	0.0 (ref.)	32.4	18.4	0.341

Configuration	MFU (%)	vs. Baseline (pp)	Comm. Overhead (%)	Pipeline Bubble (%)	TEI
+ TATP only	55.4 (3.6)	+9.2	24.8	17.6	0.398
+ TATP + DPS	59.6 (3.2)	+13.4	22.4	9.2	0.432
+ TATP + DPS + HERMP	61.0 (3.0)	+14.8	21.8	9.0	0.456
ADGT Full (all four)	62.8 (2.8)	+16.6	21.6	8.8	0.482









5. Discussion

5.1 Adaptive vs. Static Parallelism Configurations

The 40.4% MFU improvement of ADGT over manually- tuned Megatron-LM confirms the central ADGT thesis: optimal distributed training configurations are hardware-topology-specific, workload-specific, and training-phase-specific, and cannot be achieved by static configurations optimised for idealised assumptions. The TATP topology-aware tensor parallelism provides the largest individual MFU improvement (+9.2pp at 70B scale), reflecting that mismatched tensor parallelism degree is the most common efficiency failure mode in practice -- practitioners typically choose TP based on model architecture alone without considering the bandwidth asymmetry between NVLink intra- node and InfiniBand inter-node communication. The DPS pipeline bubble reduction (18.4% to 9.2% -- 50% improvement) confirms that the standard 1F1B schedule with fixed micro-batch count is significantly suboptimal for heterogeneous hardware where per-stage computation time varies. The HERMP expert load balancing benefit (+14.8% vs. +13.4% without HERMP = +1.4pp incremental) is modest at 70B MoE scale but is expected to be larger for models with higher expert count (> 8 experts) where load imbalance is more severe.

5.2 Limitations and Future Research

The ADGT evaluation is conducted on A100 clusters with HDR InfiniBand; different hardware configurations (H100 with NVLink 4.0, AMD MI300X, Google TPU v5) have substantially different bandwidth and compute profiles that may shift the relative contribution of each ADGT component. The AGCC gradient compression introduces a small gradient noise increase that was not observed to affect final model quality at the 1,000-step evaluation horizon but should be monitored over full pre-training runs of hundreds

of thousands of steps. Future research should integrate ADGT with the RCLT resource-constrained training framework (Popescu et al., 2024) to provide a comprehensive training optimisation system addressing both algorithm-level (RCLT) and infrastructure-level (ADGT) efficiency. The combination of ADGT's MFU improvements with RCLT's memory savings has the potential to achieve substantially greater training accessibility than either framework alone, enabling frontier-quality model training on significantly smaller GPU clusters.

6. Conclusion

6.1 Summary

This paper proposed the ADGT distributed training framework with four adaptive parallelism components (TATP, DPS, HERMP, AGCC) and the TEI efficiency metric. Evaluated across 7B-70B model scales on 32-256 GPU clusters: ADGT achieves mean MFU = 58.4% vs. Megatron-LM 41.6% (+40.4%); wall- clock training time reduced by 27.3% on average, 28.8% at 70B/256 GPU. Communication overhead reduced from 32.4% to 21.6%; pipeline bubble from 18.4% to 8.8%; TEI 0.341 to 0.482 (+41.3%). TATP provides largest individual gain (+9.2pp MFU); DPS provides largest bubble reduction (-50%).

6.2 Recommendations

For distributed training practitioners, we recommend adopting ADGT TATP as the highest-priority efficiency improvement: topology-aware tensor parallelism selection is simple to implement as a one-time profiling step before each training run and provides the largest individual MFU gain. DPS dynamic pipeline scheduling should be adopted for all pipeline-parallel configurations (PP >= 4), where the pipeline bubble contribution to training overhead is significant. For MoE model training, HERMP provides additional load balancing benefit beyond TATP and DPS. For large-scale clusters (>= 128 GPUs) where communication is the primary bottleneck, AGCC gradient compression provides the additional communication overhead reduction needed to maintain high MFU at extreme scale. The ADGT implementation, profiling tools, and configuration search algorithm are available as open-source extensions to Megatron-LM and PyTorch FSDP. Technical details in Appendix A.

References

Alistarh, D. et al. (2017). QSGD: Communication-efficient SGD. NeurIPS, 30.

Fedus, W., Zoph, B., and Shazeer, N. (2022). Switch transformers. JMLR, 23(120), 1-39.

Huang, Y. et al. (2019). GPipe: Efficient training of giant neural networks. NeurIPS, 32.

Karimireddy, S. P. et al. (2019). Error feedback fixes SignSGD and other gradient compression schemes. ICML 2019.

Korthikanti, V. et al. (2022). Reducing activation recomputation in large transformer models. MLSys 2023.

Lepikhin, D. et al. (2021). GShard: Scaling giant models with conditional computation. ICLR 2021.

Lian, X. et al. (2017). Can decentralized algorithms outperform centralized algorithms? NeurIPS, 30.

Lin, Y. et al. (2018). Deep gradient compression: Reducing the communication bandwidth for distributed training. ICLR 2018.

Muller, H., Horvath, P., and Dubois, E. (2025). Synthetic data generation for privacy-preserving ML. Journal of Generative Intelligence, 2025(3), 41-48.

Narayanan, D. et al. (2021). Efficient large-scale language model training on GPU clusters. SC 2021.

Popescu, L., Horvath, M., and Novak, I. (2024). Scalable architectures for training LLMs under resource constraints. Journal of Generative Intelligence, 2024(1), 1-8.

Rajbhandari, S. et al. (2020). ZeRO: Memory optimizations toward training trillion parameter models. SC 2020, 1-16.

Shoeybi, M. et al. (2019). Megatron-LM: Training multi-billion parameter language models. arXiv:1909.08053.

Silva, E., and Rossi, L. (2025). Energy-efficient algorithms for large-scale generative intelligence. Journal of Generative Intelligence, 2025(2), 17-24.

Touvron, H. et al. (2023). LLaMA 2: Open foundation and fine-tuned chat models. arXiv:2307.09288.

Dao, T. et al. (2022). FlashAttention: Fast and memory-efficient exact attention. NeurIPS, 35.

Rajbhandari, S. et al. (2022). DeepSpeed-MoE: Advancing mixture-of-experts inference and training. ICML 2022.

Ren, J. et al. (2021). ZeRO-Offload: Democratizing billion-scale model training. ATC 2021.

Lindberg, L., Hansen, E., and Ivanov, E. (2024). Efficient fine-tuning strategies for domain-specific LLMs. Journal of Generative Intelligence, 2024(1), 9-16.

Costa, O., Schmidt, I., and Costa, L. (2024). Continual learning frameworks for long-lived generative AI systems. Journal of Generative Intelligence, 2024(1), 17-24.

ISSN: e-ISSN: 3117-6429 / p-ISSN: 3117-6437 Vol: 2025, Issues: 4, 2025

Declarations

Funding

This research was supported by the Estonian Research Council under grant PRG2946 (ADGT: Adaptive Distributed Generative Training), the Austrian Science Fund (FWF) under grant P 43618-N, and the Swiss National Science Foundation (SNSF) under grant 200021-226418. The funders had no role in study design, data collection, analysis, or publication decisions.

Conflict of Interest

The authors declare no competing financial interests or personal relationships that could have influenced the work reported in this paper.

Data Availability Statement

The ADGT implementation as extensions to Megatron-LM and PyTorch FSDP, profiling tools, configuration search algorithm, and all training efficiency logs are available as open-source resources.

Ethical Approval

This study involved computational experiments only. No human subjects or personal data were involved. Ethical approval was not required.

ISSN: e-ISSN: 3117-6429 / p-ISSN: 3117-6437 Vol: 2025, Issues: 4, 2025

Appendix A

ADGT Implementation Guide and TEI Calculation

The following provides ADGT component implementation specifications and TEI calculation methodology.

TATP and DPS Implementation

HERMP and AGCC Configuration

TEI Calculation and Configuration Search