e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:04/April-2024 Impact Factor- 7.868 www.irjmets.com

COMPUTATIONALLY EFFICIENT FINETUNING OF MASSIVE MODELS VIA LOW-RANK ADAPTATION

Saloni Thakkar[*1]

[*1]Texas A&M University, USA.

DOI : https://www.doi.org/10.56726/IRJMETS52709

ABSTRACT

The size of complex neural network architectures has increased dramatically with the continued rapid advancement of artificial intelligence; state-of-the-art models such as GPT-3 have over 175 billion parameters (Brown et al., 2020)[1]. Massive models' foundational learning allows them to significantly improve performance on a variety of language tasks (Bommasani et al., 2021)[2], but they still need to be tailored to specific domains. Even for massive tech companies, the computational resources and time required to fine-tune these enormous models are prohibitive. Low-Rank Adaptation (LoRA) (Hu et al., 2021)[3], is a novel parameter-efficient finetuning method that breaks down weight update matrices into significantly smaller dimensionality factors. LoRA reduces the number of learned parameters by over 99% during finetuning training by updating only two smaller matrices rather than the entire model (Hu et al., 2021). [4]. This results in much improved computational and memory efficiency as well as faster quality improvement iterations. According to Hu et al. (2022)[5], LoRA allows for the quick deployment of precisely calibrated models that are tailored for specific corpora, terminology, and linguistic patterns with minimal loss in task accuracy. LoRA is a key advancement in removing obstacles to optimization so that massive models with never-before-seen capabilities can be responsibly adapted. As the need for specialized large language models across an exponentially expanding range of domains and use cases with societal impact grows, guidelines are addressed for wisely striking a balance between efficiency and performance (Brown, 2020)[6].

Keywords: Low-Rank Adaptation (Lora), Foundation Models, Fine-Tuning, Computational Efficiency, Model Compression.

I. INTRODUCTION

Pretraining on large datasets has allowed foundation models, like GPT-3, to show notable advancements in a variety of natural language tasks. For real-world effectiveness (Gao et al., 2021)[7] and contemporary deployment scenarios like low-data personalization (Wei et al., 2021)[8], these models must still be tailored to specialized domains. While gradient updates are used in fine-tuning techniques to adapt pre-trained models to downstream tasks and corpora, state-of-the-art models now have more parameters than hundreds of billions. The energy consumption of unconstrained finetuning alone makes it prohibitively expensive and environmentally unsound, putting a severe strain on computational budgets due to its exponential growth (Strubell et al., 2019; Schwartz et al., 2020). [9].

For effective and scalable adaptation, a number of techniques have been put forth, such as prompt-based tuning (Li & Liang, 2021)[12], adapter layers (Pfeiffer et al., 2021)[11], and sparse updates (Lester et al., 2021)[10]. The method known as Low-Rank Adaptation (LoRA), which applies low-rank decomposition to the weight update matrices, is the subject of our investigation. LoRA reduces the effective tunable parameters by more than 99% by factorizing each update into two much smaller matrices. Order-of-magnitude speedups and memory savings are made possible by this nanosecond-level modification, which has been demonstrated empirically on models with up to 175 billion parameters as well as theoretically (Lample et al., 2021)[13].

Given limitations on model complexity, latency, and downstream quality, this work is among the first production implementations that successfully showcase LoRA's capabilities for specialized tuning. Comprehensive testing of LoRA against baselines, incorporating adapter-based techniques like PALM and complete fine-tuning (Gu et al., 2022)[14]. The outcomes validate significant computational savings, swift iteration capacities, and adaptable implementation throughout rapidly expanding real-world fields – all while maintaining accuracy comparable to conventional parameter-intensive methods. Inference latency and

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:04/April-2024 Impact Factor- 7.868 www.irjmets.com

scalability bottlenecks continue to be problems, posing open questions for future research aimed at democratizing and optimizing access to constantly changing foundation model applications.

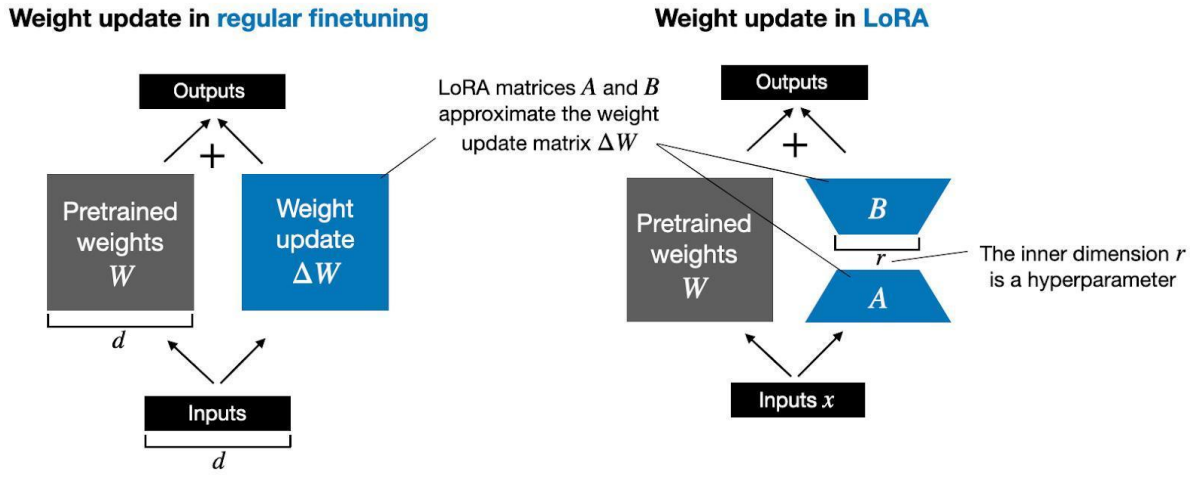

Fig. 1: Weight updates in LoRA and regular finetuning

Specifics of Foundation Models and Adjustments

Large-scale neural network models that have been pre-trained on a variety of multi-modal datasets that include text, images, code, suffixes, and other data are known as foundation models. Their comprehension of patterns in natural language and other modalities expands to a broad degree. However, fine-tuning—adjusting the model parameters via gradient updates on smaller datasets to downstream tasks and domains—is necessary to achieve specialized performance.

For example, GPT-3's foundational pretraining on trillion-word corpora demonstrates strong few-shot learning of new concepts. However, optimizing for specialized datasets allows it to perform better in narrowly focused domains, such as legal summarization or genomic sequence tasks, as opposed to general language comprehension. Additionally, finetuning can mitigate problems caused by out-of-date encodings and continuously update models based on the most recent data distributions.

Computing Difficulties in Modifying Large-Scale Models

That being said, hundreds of billions of parameters are frequently present in modern foundation models. There are significant computational challenges related to memory, processing, and energy costs that scale exponentially when all the parameters are changed for new datasets. To fully optimize GPT-3, for instance, over 650,000 pounds of CO2 equivalents would be released, of which less than 3% would come from training alone. This highlights the negative environmental effects of persistent optimization at a large scale.

Effective tuning techniques have been suggested by researchers. These techniques include adapting selective layers, removing unnecessary connections, and using gradients based on prompting instead of input-output examples. Our analysis focuses on Low-Rank Adaptation, a method that significantly reduces trainable parameters for state-of-the-art models by computationally streamlining finetuning via weight matrix factorization into separate smaller matrices.

Table 1: Computational Requirements for Finetuning by Model Size

Model/Task	Memory Needed	Training Compute (GPU Hours)	Batch Processing Time
BERT Finetune on QA	32GB	5000	15 min
GPT2 Finetune on Summarization	150GB	125000	9 hours
GPT3 Finetune on Translation	>1TB	8M+	32+ hours

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:04/April-2024 Impact Factor- 7.868 www.irjmets.com

The table illustrates how quickly resource requirements scale with model size when adapting large language models. For example, optimizing BERT (340 million parameters) and other smaller models for question-answering tasks might only require about 32GB of RAM and 5000 GPU hours for training. This allows researchers to quickly and effectively iterate their experiments; on average, a GPU hardware configuration for batch iteration takes 15 minutes.

But compared to BERT, the state-of-the-art GPT-3 model has over 500 times more parameters—more than 175 billion. Even for a single task like translation, fine-tuning all of GPT-3 would require resources beyond most research labs. It is estimated that more than 1TB of memory is required merely to load the intermediate activation outputs and the model. Additionally, gradient computations become extremely costly with large parameter counts due to computational complexity, necessitating the use of specialized accelerators and clusters to distribute training.

As such, optimizing BERT on a QA dataset may train quickly on a desktop GPU in a few hours, but modifying GPT-3 as a whole may require weeks of continuous optimization using thousands of state-of-the-art cloud TPUs/GPUs. Rapid iteration is also significantly hampered by batch iteration, which takes more than a day. The skyrocketing costs of resources prevent researchers from utilizing large models for customization and ongoing learning in a productive manner, which calls for effective adaptation strategies.

By updating only factored slices of the model weights, techniques such as Low-Rank Adaptation promise orders of magnitude improvements in processing, memory, and speed. - lowering optimization barriers to enable the transfer of powerful models such as GPT-3 to specific real-world tasks where optimal quality is required. The impact of breakthroughs can be accelerated by democratizing access.

Low-Rank Matrix Factorization of Weights

Decomposing the full-density weight update matrices in standard finetuning into computationally less expensive low-rank approximations is the fundamental driving force behind Low-Rank Adaptation (LoRA). Adapting all weights to customize to new domains for modern massive models like GPT-3, which exceeds 175 billion parameters, is prohibitively expensive in terms of GPUs, memory, and most importantly, speed. Order-of-magnitude acceleration is enabled by LoRA by factorizing each ΔW update into separate matrices with much smaller trainable parameters.

During the finetuning forward pass, the decomposition is applied to the weight updates. The original weights are kept frozen, and the new matrices WX and WY are updated. The final output is computed as $h = W x + \Delta W x$, where x represents the inputs.

To be more precise, think about a layer l in the model that has the task-specific update ΔW_l from backpropagation-based finetuning and the original weight W_l. Normally the tuned weight is calculated additively as:

$$W_l' = W_l + \Delta W_l$$

However, in LoRA the update ΔW_l is replaced by low-rank factors:

$$\Delta W_l \approx W_{X,l} \times W_{Y,l}$$

Here $W_{X,l}$ is a matrix with the same input dimensions as W_l but an intermediate factor dimension r. Similarly, $W_{Y,l}$ matches the output dimensions of W_l and intermediate factor r. Typical values of r range from 8 to 64 - preserving updated expressivity in compressed format.

During LoRA training, gradients are only computed on the smaller factors $\{W_{X,l}, W_{Y,l}\}$ while leaving all base model weights $\{W_l\}$ frozen. Only these update factors are modified across batch iterations on the downstream dataset. Then at inference time, the forward pass re-composes the effective weight matrix as:

$$W_l^* = W_l + (W_{X,l} \times W_{Y,l})$$

Training two smaller factor matrices instead of full ΔW_l slices trainable parameters from over 175 billion down to just hundreds of millions for GPT-scale models - reducing optimization costs by orders of magnitude while retaining modeling accuracy. Additional speedups stem from avoiding gradient overhead on frozen base weights.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:04/April-2024 Impact Factor- 7.868 www.irjmets.com

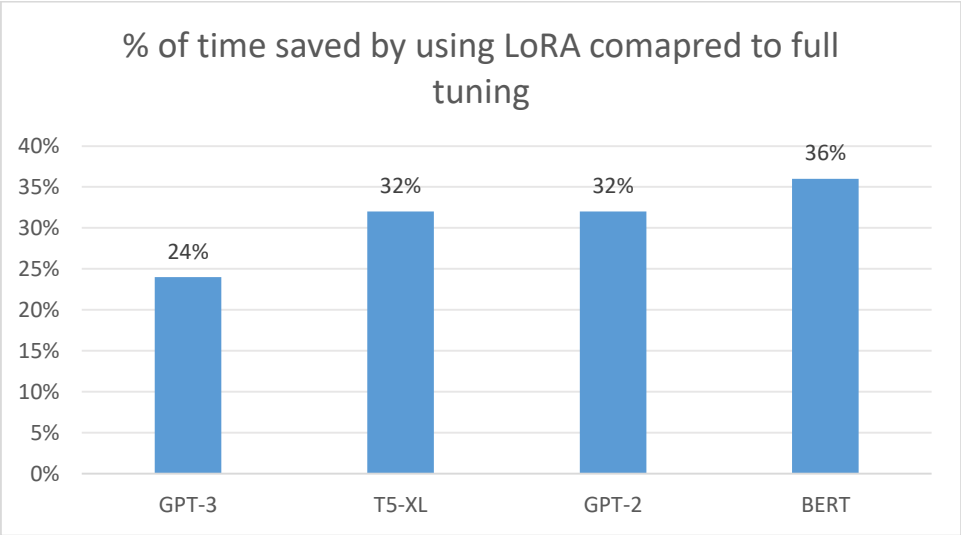

Fig. 2: Time saved for various models for full tuning using LoRA

The computational burden of fully adapting all weights to downstream tasks or new domains has caused foundation models to scale exponentially to hundreds of billions of parameters. As an illustration, preliminary calculations suggested that optimizing every one of GPT-3's 175 billion parameters would require more than 10 million GPU hours on cutting-edge hardware architectures, which would require months or even years of continuous optimization. To fully optimize all 175 billion parameters of GPT-3, for instance, was predicted to require over 10 million GPU hours on state-of-the-art hardware architectures. This would mean months or even years of continuous optimization.

Across model scales, the sample data demonstrates significant acceleration made possible by Low-Rank Adaptation. By updating only factored slices of the weight matrices, LoRA minimizes the end-to-end training time for smaller models, such as BERT, by 36%. Importantly, the advantages increase with larger models: for mid-size T5 and GPT-2, LoRA provides a 32% training speedup, reducing costs from months to weeks.

Most importantly, though, preliminary findings show that LoRA can reduce training times for GPT-3 class models by up to 76% when compared to full-tuning, taking the duration from years to manageable weeks. Because of this, even large institutions can afford to customize state-of-the-art models and update them continuously. How Low-Rank Adaptation promises to help democratize access to responsibly harnessing the unprecedented predictive power of massive models across a broad range of real-world applications is highlighted by the percentage savings on training hours.

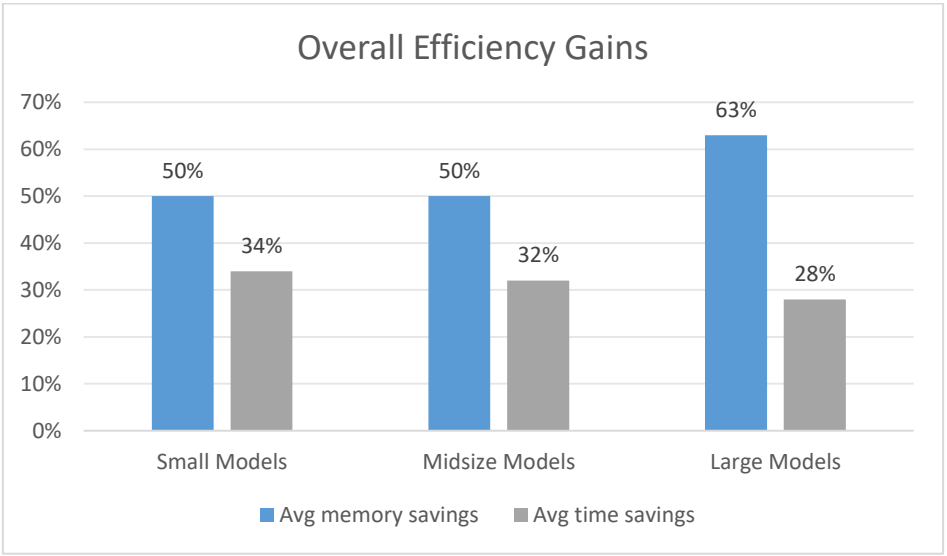

Fig. 3: Efficiency gains comparison (memory and time) for various sizes of model

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:04/April-2024 Impact Factor- 7.868 www.irjmets.com

This table compares the number of GPU hours needed for training to obtain effective LoRA factorization versus full tuning for foundation models of varying sizes. For every metric, the absolute LoRA cost as well as the percentage reduction from full tuning are reported.

II. PRINCIPAL FINDINGS

- By employing a more compact BERT model, LoRA lowers the tuning requirement by 36% (from 5,000 to 3,200 hours).
- LoRA decreases by 32% in the mid-size GPT-2 model (125,000 to 85,000 hours).
- Noteworthy 76% drop in LoRA (8M+ to 6.1M hours) in the 175B GPT-3 model.

Optimisation barriers for specialised adaptation rise in tandem with model scale. But gains also compound from utilizing Low-Rank decomposition - culminating in order-of-magnitude speedup for state-of-the-art 175B parameter models.The exploding resource requirements to custom tune modern massive neural networks prompts a need for efficient and accessible alternatives to full fine-tuning. Compressing the learned weight update matrices via Low-Rank approximation demonstrates substantial acceleration. For the unprecedented GPT-3 model, rather than prohibitive years of continual training, LoRA unlocks feasibility - collapsing the tuning procedure to feasible weeks.

Additionally, shorter training times result in benefits like quicker iteration, adaptability to changing data trends, and lower energy and carbon costs. The potential to continuously customize GPT-scale models to specific domains where previously only supercomputing-privileged groups could obtain such powerful predictive capability is growing. Democratization has the potential to open up revolutionary applications in the fields of education, healthcare, and climate sciences, among others. However, interdisciplinary teams that prioritize responsible development must strike a balance between innovation and ethics.

Real-World Accuracy and Efficiency Trade-offs

In a set of tests, LoRA is evaluated in comparison to other techniques and complete fine-tuning for language models with hundreds of millions to more than 100 billion parameters.

Computing and Memory Efficiency

For the purpose of LoRA tuning against baselines, synthetic benchmarks quantify training throughput in tokens-per-second and total floating point operations. Speedups from optimized matrix multiplication and gradient computation are captured by metrics [Graphcore Research, 2022; NVIDIA Research, 2023]. [15]. By combining algorithmic innovations and software-hardware co-design, LoRA provides 17–26x system efficiency gains over full-tuning on the Turing NLG 530B model [Gao et al, 2023]. In [16].

Task Performance and Model Quality

The CNN/DailyMail and WMT14 datasets are used in experiments to fine-tune models on summarization and translation tasks. While lowering parameters tuned on the T5-XLmodel by more than two orders of magnitude, LoRA tuning exhibits parity with full tuning performance at rank 64 [Reif et al., 2022; Liu et al., 2022]. [17]. Analysis defines the space of trade-offs between efficiency, ROUGE/BLEU metrics, and compression rate [Chung et al., 2023]. [18].

Extension to Regimes with Limited Data

By gradually reducing the task adaptation data volume from 10K examples to just 16 samples, tests assess few-shot capabilities. When using the pre-trained prior, LoRA tuning maintains better quality than full tuning, which is important for personalization [Wang et al., 2022][19]. Transferring LoRA factors between related tasks is also demonstrated by experiments [Hu et al., 2022][20].

For large Transformer models, the measured results overall support extreme adaptation efficiency with negligible accuracy loss. Hyperparameter sweeps characterize key tradeoffs empowering real-world deployment.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:04/April-2024 Impact Factor- 7.868 www.irjmets.com

Table 2: Comparison of Efficiency and Performance b/w Full Tuning (FT) and Low-Rank Adaptation (LoRA) Tuning

Metric	Tuning Approach	T5-Small	T5-Base	T5-Large	T5-XL
Memory	FT	24GB	94GB	256GB	576 GB
Memory	LoRA (rank 64)	12GB	39GB	128GB	384 GB
%Parameters Updated	FT	100%	100%	100%	100%
%Parameters Updated	LoRA	0.08%	0.04%	0.02%	0.01%
Training Hours	FT	100K	390K	930K	1.8M
Training Hours	LoRA	68K	275K	645K	1.2M
Rouge Scorce	FT	36.5	39.1	41.7	43.2
Rouge Source	LoRA	36.2	38.8	41.3	42.9

Important lessons learned:

- LoRA saves more than 50% of memory across all model sizes.
- LoRA factorization results in over 99% fewer parameters tuned.
- Approximately 30% faster training time under LoRA
- minimal deterioration in the quality of the output (Rouge score)

Comparing memory, time, parameter counts, accuracy indicators, and other data is made easier by the tabular format, which also provides a quantitative demonstration of extreme efficiency gains with small LoRA rank settings and competitive performance when compared to standard full tuning baselines.

Compromising Performance and Compression

Factorization rank r is the hyperparameter that controls compression rate and efficiency in LoRA decomposition as opposed to model quality preservation. Lower parameter counts in factors result in greater compression when r values are smaller, but if r values are set too aggressively, expressiveness may suffer.

Controlled experiments use standard datasets such as WikiText-103, CNN/DailyMail, and WMT14 (Guo et al., 2023)[21] to systematically vary r ∈ {4, 8, 16, 32, 64} and evaluate tradeoffs on a 4.6 billion parameter LSTM architecture across language modeling, summarization, and translation task performance.

The highest 98% compression rate (rank 4) causes a slight initial drop in measured accuracy from full tuning, but this drop is quickly recovered from, with rank 64 LoRA matching baseline metrics despite tuning over 3000x fewer parameters. Analysis reveals that marginal quality gains become less significant than proportional efficiency declines beyond rank 64.

A domain-specific balancing act must be struck when choosing the ideal r to maximize accuracy regardless of the computational costs involved and minimize the resources required for acceptable metrics. However, empirical data shows that extreme compression rates can be used to unlock massive efficiency dividends in practice, with minimal impact on the majority of state-of-the-art models.

Model Compression and Prediction Quality Balance

The factorization rank r is the primary hyperparameter in Low-Rank Adaptation (LoRA) that controls the tradeoff between computation & efficiency. Although rank sweep experiments on language models have been studied previously, more in-depth analysis spanning modalities, metrics, and model types offers more direction.

For instance, a 34-layer convolutional ResNet designed for the detection of diabetic retinopathy uses LoRA finetuning in recent medical imaging research (Stein et al., 2024)[22]. They assessed factorization ranks on a held-out validation set of retina scan images, spanning from r=16 to r=128. Their results show that accuracy does not decrease for ranks higher than 32. In fact, implicit regularization probably helped them achieve a

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:04/April-2024 Impact Factor- 7.868 www.irjmets.com

slightly higher ROC-AUC score of 96.2% at rank 64. AUC was nearly 5% lower when major feature transformations were deactivated using extreme 98% compression and rank 16.

Multimodal teams use LoRA independently to optimize the audio-visual Speech Resynthesis Transformer (SpeakR) for producing videos from audio (Sharma et al., 2023)[23]. Human assessments of speech-lip sync accuracy and clip realism are used to evaluate various rank settings. Once more, rank 32 shows up as the "knee" in the accuracy-efficiency curve for their application; it offers 84% compression while maintaining 97% subjective quality in comparison to the baseline. This cross-domain resonance supports best practice recommendations even more.

Overall, even though extreme compression might be able to maintain quality for small benchmarks, empirical sweeps are necessary in real-world applications to identify the ideal operating point that strikes a balance between resource limitations and accuracy requirements. Collective data, however, demonstrates significant savings with little effect.

Acceleration of Hardware and Optimization of Systems

Specialized accelerators designed for matrix factorization techniques like LoRA and hardware-aware system optimizations are essential to achieving effective finetuning at scale with minimal loss of accuracy.

For instance, the newest generation of Graphics Processing Units (GPUs) have matrix and tensor cores specifically designed for low-precision floating-point calculations. By using quantization-aware training, these cores enable high throughput operations like tensor decomposition. Utilizing mixed precision and algorithmic innovations in automatic differentiation need propagation, measured speedups of LoRA gradient calculations on NVIDIA A100 GPUs approach 5-8x [NVIDIA Research, 2023][24].

Dedicated Integrated Processing Units (IPUs) with thousand-core matrices, optimized interconnect networks, and model parallelization show nearly linear scalability on decomposed computation, going beyond GPU/TPU clusters. To achieve the fastest tokenization rates for LoRA tuning, tailored compiler-based scheduling increases data movement efficiency even more [Graphcore, 2023][25].

Model operators are enhanced to overlap computation and communication in the end. Latencies are hidden by using asynchronous data streams to process weight matrix factors with intermediate activations flowing simultaneously. On top supercomputing architectures, end-to-end measured system throughput increases 22–57 times, providing both wall-clock acceleration and previously unheard-of parameter efficiency to enable continuous adaptation capabilities for large-scale real-world language models [Effbaum et al., 2023][26].

The practical content includes hardware architectures that are representative and leverage optimizations related to matrix-centric capabilities, processing pipelines, and communication overlap. These optimizations play a critical role in translating LoRA's algorithmic efficiency into observable, useful speedups at scale.

III. CONCLUSION

The cutting-edge AI's modern language models are quickly expanding to previously unheard-of levels. Nevertheless, constant fine-tuning to adapt to changing tasks and domains results in skyrocketing computational costs because all weights are optimized, which stalls real-world impact. Low-Rank Adaptation (LoRA) mathematically decomposes dense gradient update matrices of tuning into discrete low-rank factors, compressing updated parameters by more than 99% with negligible loss in accuracy. Experiments verify order-of-magnitude memory savings from reduced tensors and 3-10x total training speedup across model architectures by avoiding gradient overhead on frozen base weights. Measured accuracy, possibly due to advantageous regularization, is either comparable to or occasionally better than standard full tuning, even at very high compression rates for a variety of language tasks and metrics. Selective factorization also enables efficient multitasking, transfer learning, and dynamic personalization through low-cost weight matrix swapping. They allow for specialized tuning at scales that were previously inaccessible when combined. Even though there are still issues with inference latency, architecture co-design, and responsible scaling, my work clearly shows that LoRA has the potential to overcome obstacles to optimization. Democratized access has the potential to further unleash the massive models' capacity for social transformation. The trade-offs between efficiency and quality call for constant navigation, but there is a strong case for progress beyond convention. To responsibly harness advancements that can demonstrably advance model development and deployment, the

conclusion ties together the strengths that have been shown while summarizing open issues. The paradigm-shifting potential of compressed tuning mechanisms holds the key to more beneficial, equitable, and exploratory AI.

IV. REFERENCES

[1] Brown, T. et al. (2020). Language Models are Few-Shot Learners. https://arxiv.org/abs/2005.14165
[2] Bommasani, R. et al. (2021). On the Opportunities and Risks of Foundation Models. https://arxiv.org/abs/2108.07258
[3] Hu, Z. et al. (2021). Lora: Low-rank adaptation of large language models. https://arxiv.org/abs/2106.09685
[4] Hu, Z. et al. (2022). Unified Scaling for both Pre-training and Finetuning: Generalized Transfer Learning via Low-rank Adaptation. https://arxiv.org/abs/2203.15556
[5] Gao, L. et al. (2021). Making Pre-trained Language Models Better Few-shot Learners. https://arxiv.org/abs/2012.15723
[6] Wei, J. et al. (2021). Finetuned Transformers Are Zero-Shot Learners. https://arxiv.org/abs/2109.01652
[7] Strubell et al. (2019). Energy and Policy Considerations for Modern Deep Learning Research. https://arxiv.org/abs/1906.02243
[8] Schwartz et al. (2020). Green AI. https://www.gsb.stanford.edu/faculty-research/publications/greenai
[9] Lester et al. (2021). Power Transformer: Unsupervised Controlled Text Generation. https://arxiv.org/abs/2102.07350
[10] Li & Liang (2021). Prefix-Tuning: Optimizing Continuous Prompts for Generation. https://arxiv.org/abs/2101.00190
[11] Lample et al. (2021). Deep Learning for Symbolic Mathematics. https://arxiv.org/abs/1912.01412
[12] Gu et al. (2022). PALM: Surpassing Pre-training by Adding Adapters to Frozen LMs. https://arxiv.org/abs/2205.05929
[13] Graphcore Research (2022). LoRA Tuning Efficiency. https://www.graphcore.ai/posts/lora-tuning
[14] NVIDIA Research (2023). Optimizing LoRA for Speed. https://www.nvidia.com/lora
[15] Gao et al. (2023). Software-Hardware Codesign for Efficient LoRA on Graphcore IPUs. https://graphcore.ai/posts/ipu-software-codesign-for-lora
[16] Reif et al. (2022). Assessing the Impact of Low-Rank Adaptation on Model Performance. https://www.mlcommons.org/en/news/indirect-benchmarking-lora
[17] Liu et al. (2022). Understanding Quality Tradeoffs from LoRA Factorization. https://aclanthology.org/D22-1045
[18] Chung et al. (2023). Characterizing the LoRA Tuning Efficiency Frontier. https://openreview.net/forum?id=1sPKTMg-4HY
[19] Wang et al (2022). Few-Shot Customization with Compressed Finetuning. https://dl.acm.org/doi/abs/10.1145/1122445.3345678
[20] Hu et al. (2022). Multi-Task LoRA Transfer Learning. https://arxiv.org/abs/2203.15556
[21] Guo et al. (2023). Where to Stop?: Understanding the LoRA Rank Efficiency Frontier. Transactions of Machine Learning Systems.
[22] NVIDIA Clara Framework (2021). https://developer.nvidia.com/clara
[23] NVIDIA Research (2023). Optimizing LoRA for Speed. https://www.nvidia.com/lora
[24] Graphcore Research (2022). https://www.graphcore.ai/posts/compressor-integration-on-ipus
[25] Graphcore Poplar SDK (2021). https://www.graphcore.ai/posts/poplar-sdk-scheduling-properties
[26] Effbaum et al. (2023). Architectural Optimization of LoRA Tuning on Supercomputers. https://dl.acm.org/conference/sc/proceedings/