Scaling AI Sustainably

LLMs, Fast and Everywhere: Acceleration for the Age of GenAI

Carole-Jean Wu

Director Meta FAIR

Keynote ISCA ARC-LG Workshop: New Approaches for Addressing the Computing Requirements of LLMs and GNNs

Buenos Aires, Argentina

June 30, 2024

Decades of Innovations in Computer Systems







>1,000,000x+Transistors

O(10k) GPUs

Domain-Specific Acceleration

750x Faster

Vertical Scaling

Horizontal Scaling

Specialization

Efficiency Keeps Computing's Energy Growth ~O(1)



Computing Industry Faces an Unprecedented Growth



Carole-Jean Wu

Computing's Energy Footprint





Al's Operational CO₂e

 CO_2e (kg)



AI model development lifecycle

- Offline training
- Online training
- Inference

Sustainable AI: Environmental Implications, Challenges and Opportunities. Wu et al. MLSys-2022.

🤰 You

I am planning a 5-day trip to Buenos Aires Argentina, mostly for the ISCA conference with some personal touring opportunities. any suggestions for spots to visit, restaurants?

🔘 Meta Al

Buenos Aires is a vibrant city with a rich culture and history. Here are some suggestions for spots to visit and restaurants to try:

Must-visit spots:

- 1. Plaza de Mayo: The heart of Buenos Aires' political and social life.
- 2. Recoleta Cemetery: A stunning necropolis with grand mausoleums.
- 3. La Boca: Colorful neighborhood with street art, tango performances, and football passion.
- 4. Teatro Colón: One of the world's most famous opera houses.
- 5. San Telmo Market: Antique shops, street food, and live music.

Restaurant suggestions:

- 1. Don Julio (Parrilla): Try their famous asado (Argentine BBQ).
- 2. El Cuartito (Pizzeria): Delicious pizza and empanadas.
- 3. La Parrilla de la Renaissance (Steakhouse): High-quality meat and wine.
- 4. Café Tortoni (Café): Historic spot for coffee and pastries.
- 5. El Desnivel (Bistro): Cozy spot for breakfast and brunch.

Additional tips:

- Try some local street food, like choripan (grilled sausage sandwich) or churros.
- · Explore the street art and murals in La Boca and San Telmo.
- Take a tango lesson or watch a live performance.
- Visit the Museo Nacional de Bellas Artes (Fine Arts Museum) for an impressive art collection.

Packing list:

Based on your trip details, I'll create a packing list with essentials for a conference, sightseeing, and enjoying the local culture. Here's a sneak peek:

- · Business attire for the conference
- · Comfortable walking shoes
- · Power adapter for Argentina (Type C and I)
- · Sun protection (sunglasses, hat, sunscreen)
- · Lightweight clothing for warm days
- · Camera for capturing street art and landmarks
- Spanish phrasebook or translation app

Let me know if you'd like me to add or modify anything on this list!



Al's Operational CO₂e



MAKE IT EFFICIENT

Understand where cycles go in your workloads

DLRMs, LLMs, Multi-Modal Generative AI Tasks

Recommendation

50% of all AI Training Cycles

80%

of all AI Inference Cycles

2012 2018 2023 Deep Learning Revolution

DLRMs, LLMs, Multi-Modal Generative AI Tasks

Recommendation

50% of all AI Training Cycles

80%

of all AI Inference Cycles

LLMs Generative AI

Better quality

Improved content classification

Conversational access to generation

2012

2018

2023

Deep Learning Revolution

https://engineering.fb.com/2024/03/12/data-center-engineering/building-metas-genai-infrastructure/

DLRMs, LLMs, Multi-Modal Generative AI Tasks



https://engineering.fb.com/2024/03/12/data-center-engineering/building-metas-genai-infrastructure/

LLMs with Distinct System Challenges

	DLRM	Llama-65B	
Training	O(100 GF/iter)	O(1 PF/sentence) la	100+x rger training clusters
Inference	O(GF/s) for 100ms latency	Prefill : O(10 PF/s) for time-to-first-token <1s	compute intensive
		Decode : Token generation <50 ms	memory intensive

Scaling Challenges for LLMs







Compute

Real-time, interactive response

O(TFLOPS) compute capacity Memory

Accelerator memory limitation in size & bandwidth **Sustainability**

Scalability TCO, OpEx, CapEx

e2e carbon footprint optimization

Excessive inference latency on CPUs (>1s)

- But ... accelerators are expensive and you may not even be able to get it!
- Maximizing accelerator utilization: HW-SW co-design
- Improving e2e efficiency: system-level optimization



How do we provide best model quality at-scale by making deployment viable?

A+++	
A++	
A +	
В	
С	
D	





Boost Efficiency with Co-Design

Re-Use "Embeddings" with Caching On-device AI

Maximize performance per W/\$/CO₂e Exploit repeated query evaluation requests by caching embeddings (MultiRay) Fast response, improved privacy. Stringent limits!

Accelerator Software/Hardware Co-Design



Reduced Precision

Reduced precision FP (FP16, BF16, F8, F6, F5, F4!) for increased compute density

Weight & Dynamic Quantization

32b ---→ 4b

Quantization reduces memory bandwidth and memory needs with higher compute density



Algorithmic Improvements

Accelerated Transformers with optimized Scaled Dot-Product Attention (SDPA + Flash Attention)

Over 10x Effective FLOPS growth (Weight-only Q.)



Accelerating Generative AI with PyTorch II: GPT, Fast, PyTorch Blog

MultiRay Caching: Re-use Processing with Embeddings

- Specialized models are costly
 - (Cost Per Inference) x (Num Content) x (Num Tasks)
- Most effort is spent on understanding the input
 - How to re-use this work for different tasks?

Embeddings!

- Input can be represented by embeddings, an intermediate result of evaluating a model
- Centralized Universal Model
 - Input: Different media (text, images, video)
 - Output: Embedding
 - Embeddings power decisions of simpler downstream models



Layer Skip: Early Exit and Self-Speculative Decoding

Faster LLM Inference

Q1: How to make model predict tokens earlier?

Prior works – training with addl. early exit heads

=> higher memory footprint; longer training time

Our proposed design – Layer dropout with increasing rates across layers *and* early exit loss shared heads

Q2: How to ensure earlier predictions are correct?

Prior works - heuristics to predict when to exit

=> increasing model memory footprint; design complexity

Our proposed design – Self-speculative decoding: decode with earlier layers, verify/correct with remaining layers

Key Insight: Layer Skip LLMs are Supernets

- Training an LLM with layerskip gives us N different models at different sizes and quality (where N == number of layers).
- Deployment and generation from these N models is also simple because they are all just transformers!

Let's use these smaller models within the LLM SuperNet to give speculative decoding from the same model.

• We term this **Self Speculation**

Self Speculation on Early Exit

- Make the draft model the first E layers of the main model
- Lower memory need
 - Do not need to store the second model
- Easier
 - Choose which draft model to use trading off accuracy and speed all by training a single model
- Faster
 - Verification can reuse the compute of the first E layers (by the draft model)





Layer Skip: Early Exit and Self-Speculative Decoding

Faster LLM Inference

	Llama2 7B						
Generation	E	d	ROUGE-2	Token Acc.	Tokens per Sec.	Speedup	
CNN-DM One-Shot Abstrac	CNN-DM One-Shot Abstractive Summarization						
Autoregressive Early Exit	- 8	-	$0.079 \\ 0.012$	-	$62.7 \\ 232.4$	1.00× -	
Self Speculative	8	12	0.078	68.9%	127.9	1.86×	
Draft and Verify	n/a	n/a					
XSUM Abstractive Sumr	XSUM Abstractive Summarization						
Autoregressive Early Exit	- 8	-	$0.073 \\ 0.002$	-	$\begin{array}{c} 63.4\\ 228.0\end{array}$	1.00×	
Self Speculative	8	12	0.073	54.6%	104.7	1.54×	
Draft and Verify	n/a	n/a					
HumanEval Coding							
Autoregressive Early Exit	- 8	-	0.041	-	$\begin{array}{r} 62.9 \\ 225.4 \end{array}$	1.00×	
Self Speculative	8	6	0.042	67.1%	122.8	1.83×	

Layer Skip: Enabling Early Exit Inference and Self Speculative Decoding. M. Elhoushi, A. Shrivastava, D. Liskovich A. Roman, A. Aly, B. Chen, C.-J. Wu. ACL-2024.

LayerSkip: Enabling Early Exit Inference and Self-Speculative Decoding

Mostafa Elhoushi^{1,†,*}, Akshat Shrivastava^{1,†,*}, Diana Liskovich^{2,†}, Basil Hosmer¹, Bram Wasti², Liangzhen Lai³, Anas Mahmoud⁴, Bilge Acun¹, Saurabh Agrawal⁶, Ahmed Roman⁷, Ahmed A Aly³, Beidi Chen^{1,5}, Carole Jean-Wu¹

¹FAIR at Meta, ²GenAI at Meta, ³Reality Labs at Meta, ⁴University of Toronto, ⁵Carnegie Mellon University, ⁶University of Wisconsin-Madison, ⁷Dana-Farber Cancer Institute *Equal Contribution, [†]Core Contributor

We present LayerSkip, an end-to-end solution to speed-up inference of large language models (LLMs). First, during training we apply layer dropout, with low dropout rates for earlier layers and higher dropout rates for later layers, and an early exit loss where all transformer layers share the same exit. Second, during inference, we show that this training recipe increases the accuracy of early exit at earlier layers, without adding any auxiliary layers or modules to the model. Third, we present a novel self-speculative decoding solution where we exit at early layers and verify and correct with remaining layers of the model. Our proposed self-speculative decoding approach has less memory footprint than other speculative decoding approaches and benefits from shared compute and activations of the draft and verification stages. We run experiments on different Llama model sizes on different types of training: pretraining from scratch, continual pretraining, finetuning on specific data domain, and finetuning on specific task. We implement our inference solution and show speedups of up to $2.16\times$ on summarization for CNN/DM documents, $1.82\times$ on coding, and $2.0\times$ on TOPv2 semantic parsing task.

Date: April 30, 2024

Correspondence: Mostafa Elhoushi, Akshat Shrivastava at melhoushi@meta.com, akshats@meta.c

Code: In progress

2024

Apr

29

CL

CS.

rXiv:2404.16710v2

 ∞

1 Introduction

Large Language Models (LLMs) have been deployed to many applications, yet their high compute and memory requirements lead to high financial and energy costs when deployed to GPU servers Samsi et al. (2023). Acceleration solutions do exist to deploy to commodity GPUs on laptops but they suffer from significant drop in accuracy Zhu et al. (2023). Accelerating LLMs further to mobile or edge devices is still an active research area Çôpli et al. (2023); Liu et al. (2024). While a large portion of LLM acceleration approaches reduce number of non-zero weights Xia et al. (2023) (a.k.a. sparsity), number of bits per weight Xiao et al. (2023) (a.k.a. quantization), number of heads per layer Shim et al. (2021) (a.k.a. head pruning), a smaller portion of approaches focus on reducing number of layers Fan et al. (2020); Elbayad et al. (2020). In this paper, we explore reducing the number of layers required for each token by exiting early during inference. Unlike quantization or sparsity, acceleration by reducing number of layers does not require specialized hardware or software kernels.

ACL-2024

Moreover, a popular research trend in LLM acceleration is speculative decoding Leviathan et al. (2023); Chen et al. (2023) that has no drop in accuracy, where a large model, referred to as the main model, is accompanied with a faster model, referred to as the draft model. The advantage of speculative decoding is that it leads to faster inference compared to the main model, but requires a larger memory footprint and complexity in implementation to maintain key-value (KV) cache in two different models. In addition to exiting early, this paper also proposes combining exiting early with speculative decoding to propose a self-speculative decoding approach that does not require an additional model or auxiliary layers.

CHAI: Clustered Head Attention for Efficient LLM Inference

Find heads whose attention score is similar!



CHAI

 <u>Key Observation</u>: A high amt. of correlation across the output of various attention heads in Multi-Head Attention – output of several attention heads focuses on same token



CHAI: 1.73x Faster Prefill and 5x Faster Decoding

Method	PIQA	HellaSwag	Arc-Challenge	Arc-Easy	BoolQ
MHA	79.8	76.1	47.5	72.8	76.0
DejaVu-10%	-3.9	-4.7	-5.78	-3.18	-7.4
DejaVu-30%	-13.3	-18.6	-18.75	-4.2	-20.2
DejaVu-50%	-24.6	-50.7	-19.35	-46.3	-21.6
SpAtten	-41.4	-42.5	-18.0	-40.2	-27.1
CHAI-static	-4.0	-4.3	-3.7	-2.5	-0.8
CHAI	-2.0	-3.2	-0.5	0.3	0.1



Time to Next Token (Decoding)

CHAI: Skewed Cluster Sizes (# of Attention Heads)



CHAI: Clustered Head Attention for Efficient LLM Inference. S. Agarwal, B. Acun, B. Hosmer, M. Elhoushi, Y. Lee, S. Venkataraman, D. Papailiopoulos, C.-J. Wu. ICML-2024.

CHAI: Skewed Cluster Sizes (# of Attention Heads)



CHAI: Clustered Head Attention for Efficient LLM Inference

Saurabh Agarwal ¹ Bilge Acun ² Basil Hosmer ² Mostafa Elhoushi ² Yejin Lee ² Shivaram Venkataraman ¹ Dimitris Papailiopoulos ¹ Carole-Jean Wu ²

Abstract

Large Language Models (LLMs) with hundreds of billions of parameters have transformed the field of machine learning. However, serving these models at inference time is both compute and memory intensive, where a single request can require multiple GPUs and tens of Gigabytes of memory. Multi-Head Attention is one of the key components of LLMs, which can account for over 50% of LLMs memory and compute requirement. We observe that there is a high amount of redundancy across heads on which tokens they pay attention to. Based on this insight, we propose C Attention (CHAI). CHAI c a high amount of correlation runtime, thus reducing both In our experiments, we show that CHAI is able to reduce the memory requirements for storing K,V cache by up to 21.4% and inference time latency by up to 1.73× without any fine-tuning required. CHAI achieves this with a maximum 3.2% deviation in accuracy across 3 different models (i.e. OPT-66B, LLAMA-7B, LLAMA-33B) and 5 different evaluation datasets.

LLMs have demonstrated remarkable performance on lan-

guage modelling tasks ranging from question answering,

text summarizing, language translation. However, such per-

formance has been achieved by scaling models to trillions of

parameters, and existing works (Hoffmann et al., 2022; Tou-

vron et al., 2023a; Kaplan et al., 2020) show that increasing

Inference on LLMs introduce several new challenges.

Beyond just the quadratic computation cost of self-

attention (Vaswani et al., 2017) with increasing context and

large model sizes, LLMs also store intermediate Key (K)

and Value (V) pairs for subsequent next word prediction.

dence to: Saurabh Agarwal <agarwal@cs.wisc.edu>.

¹University of Wisconsin-Madison ²Meta-FAIR. Correspon-

the model size may lead to even higher model quality.

1. Introduction





This K.V caching introduces additional memory related challenges as K.V cache size increases with increase in sequence length. The architecture of widely used LLMs like GPT (Brown et al., 2020) and LLAMA (Touvron et al., 2023a;b) use Multi-Head Attention (MHA) (Vaswani et al., 2017). MHA uses several attention heads to look at a sequence. As models grow bigger, the number of heads increases as well. For example, LLAMA-7B uses 32 attention heads in each layer, while LLAMA-65B uses 64 attention heads per layer (Touyron et al., 2023a). The use of MHA exacerbates bottlenecks for serving LLMs. First, it increases compute pressure due to repeated application of the attention operation. Second, it increases the memory pressure due to requiring storage of Key (K), Value (V) caches that comes with the additional attention heads. To alleviate these bottlenecks, prior works have introduced primarily two types of methods - (i) pruning of LLMs to utilize sparsity based on the input context (Liu et al., 2023b; Voita et al., 2019) and (ii) Co-designing of the Attention module to reuse components across multiple heads like Multi-Query Attention (MOA) (Shazeer, 2019) and Grouped-Ouerv Attention (GOA) (Ainslie et al., 2023).

Pruning LLMs can potentially ease the compute bottleneck, however it is challenging as the classical methods for pruning (Frankle & Carbin, 2018; Chen et al., 2020b; You et al.,

Parallelizing ML Models on Distributed Systems



Efficiently scale model training with

more optimal parallelization strategies for distributed systems

parameter shards from other GPUs before computation. ReduceScatter gradients during backward pass. activation and gradient partial sums during forward and backward pass. gradients. During backward pass, AllReduce weight gradients.

Efficiently Accelerate Model Training on Distributed Systems

MAD-Max: a performance model for model parallelization design space exploration

- Model architectures
- Training system hardware compute, memory, networking

Optimal parallelization results in an average of 65.9% (upto 2.24x) training throughput improvement



MAD-Max Beyond Single-Node: Enabling Large Machine Learning Model Acceleration on Distributed Systems. S. Hsia, A. Golden, B. Acun, N. Ardalani, Z. DeVito, G.-Y. Wei, D. Brooks, C.-J. Wu. ISCA 2024.

Efficiently Accelerate Model Training on Distributed Systems



Efficiency Optimization in O(100)

But Jevon's Paradox



Sustainable AI: Environmental Implications, Challenges and Opportunities. Wu et al. MLSys-2022.

Embodied CO₂e

An underexplored aspect of computing



Embodied CO₂e is Significant

Bringing Supercomputing Technologies On-Device

- Scale with user count
- Real-time latency
- Data ownership
- Disconnected operation
- Privacy

Designing Computer Systems for Sustainability

What does sustainability-first computing look like?

Carole-Jean Wu Meta

IBM Research

Babak Falsafi

Gage Hills Harvard University

Bobbie Manne AMD

ISCA 2024 Buenos Aires, Argentina

EPFL / SDEA

Tuesday 4:15 – 5:15pm July 2, 2024