Federation of Experts: Efficient Distributed Inference for Large Language Models

Muhammad Shahir Abdurrahman
Chun Deng, Philip Levis, Azalia Mirhoseini

DAM 2nd Summer Retreat June 2026

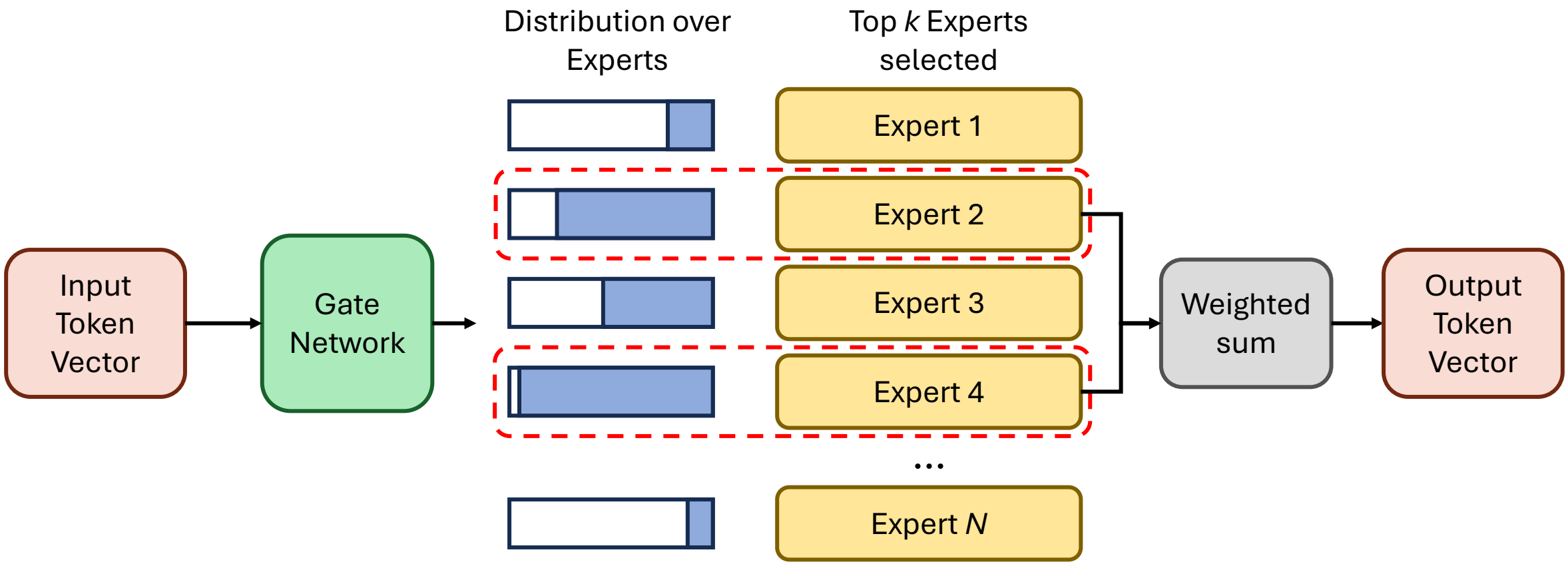
DAM

Stanford Differentiated Access Memory Project

Why Mixture of Experts?

- Large Language Models (LLMs) consistently improve as parameter counts increase.
- However, computation scales proportionally with active parameters, imposing limits on both training and inference.
- Mixture-of-Experts (MoE) architectures mitigate this by activating only a subset of the neural network per token, decoupling total parameters from active parameters.
- These efficiency gains, however, introduce new performance bottlenecks.

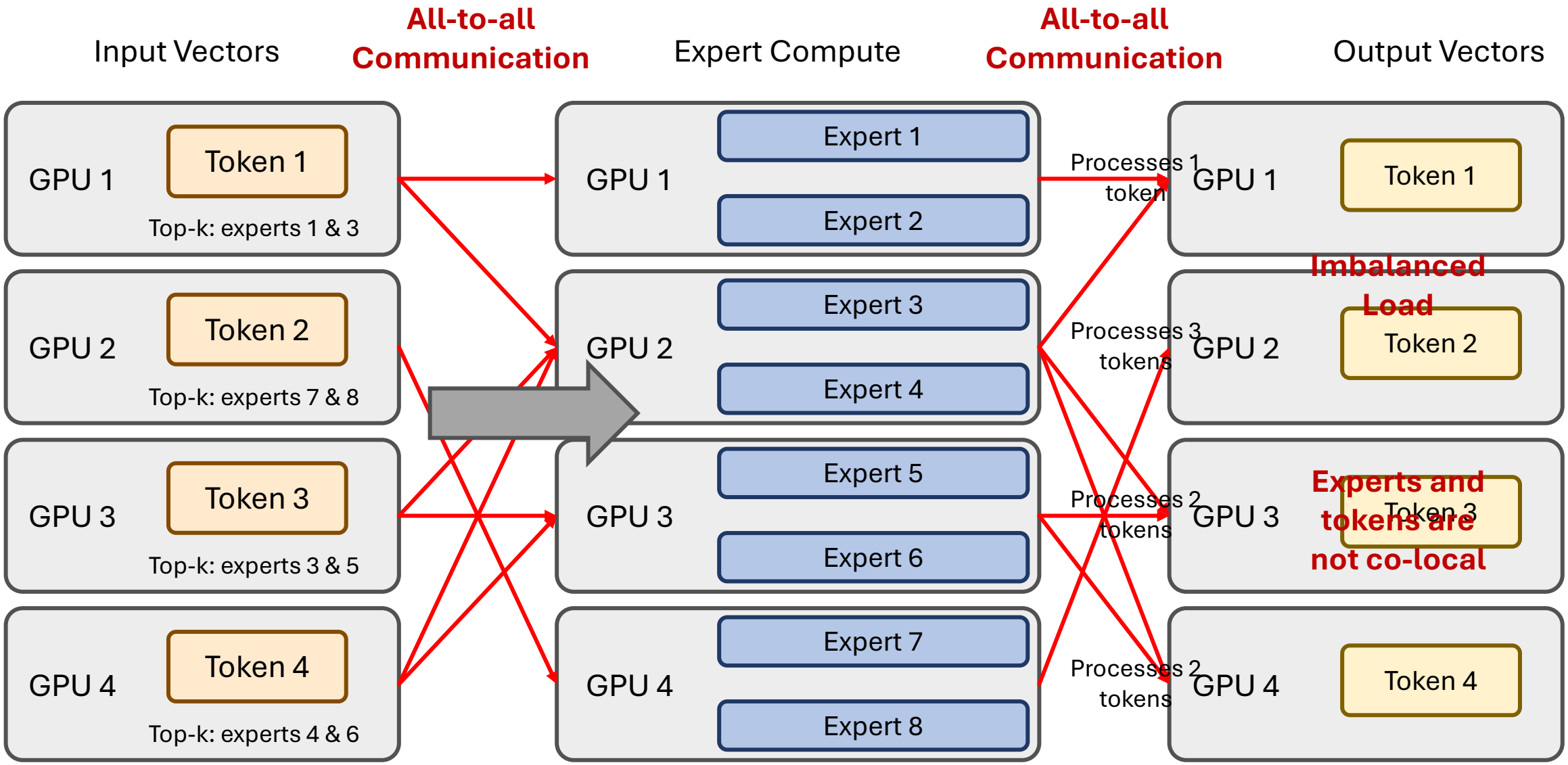
Mixture of Experts Network



Performance Challenges in Mixture of Experts

- Mixture of Expert models introduces major performance bottlenecks:
  - Load balancing of expert compute across GPUs
  - Co-location of expert parameters and tokens
  - All-to-all communication across GPUs
- Existing work focuses on faster execution but does not tackle the problem directly
  - Faster kernels for expert parallelism
  - Speculative expert-GPU placement

Distributed Mixture of Experts



Data-Compute Dependency

- There are 3 compute dependencies that enforce the all-to-all requirement in an expert parallel setup:
  1. The selection of *k* experts for each token, that can differ from layer to layer
  2. The weighted sum of the outputs from the *k* experts selected for each token
  3. The attention computation of the current token with all previous tokens which have different selected experts than the current token
- When expert parameters (memory) and tokens (compute) are no co-located, expensive bandwidth-bound communication is needed

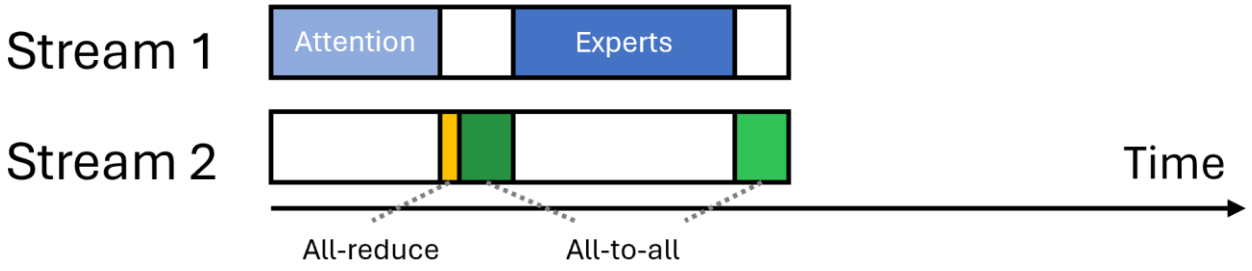
Federation of Experts

- We present the novel Federation of Experts architecture, which restructures the transformer layer to co-locating memory and compute thus reducing communication and improving load balancing.
- Compared to FoE, a standard MoE model has:
  - End-to-end latency: 1.0-5.2x
  - Time to first token: 2.6-3.6x
  - Time between tokens: 1.1-2.0x
- FoE maintains the same model quality as MoE at equivalent number of parameters.

Tackling the Performance Bottlenecks

- Federation of Experts (FoE) solves this by restructuring the MoE architecture to co-locate memory and compute
- FoE additionally uses all-reduce which saves significant bandwidth over traditional all-to-all
- Total forward-pass latency is significantly reduced

Mixture of Experts Timing

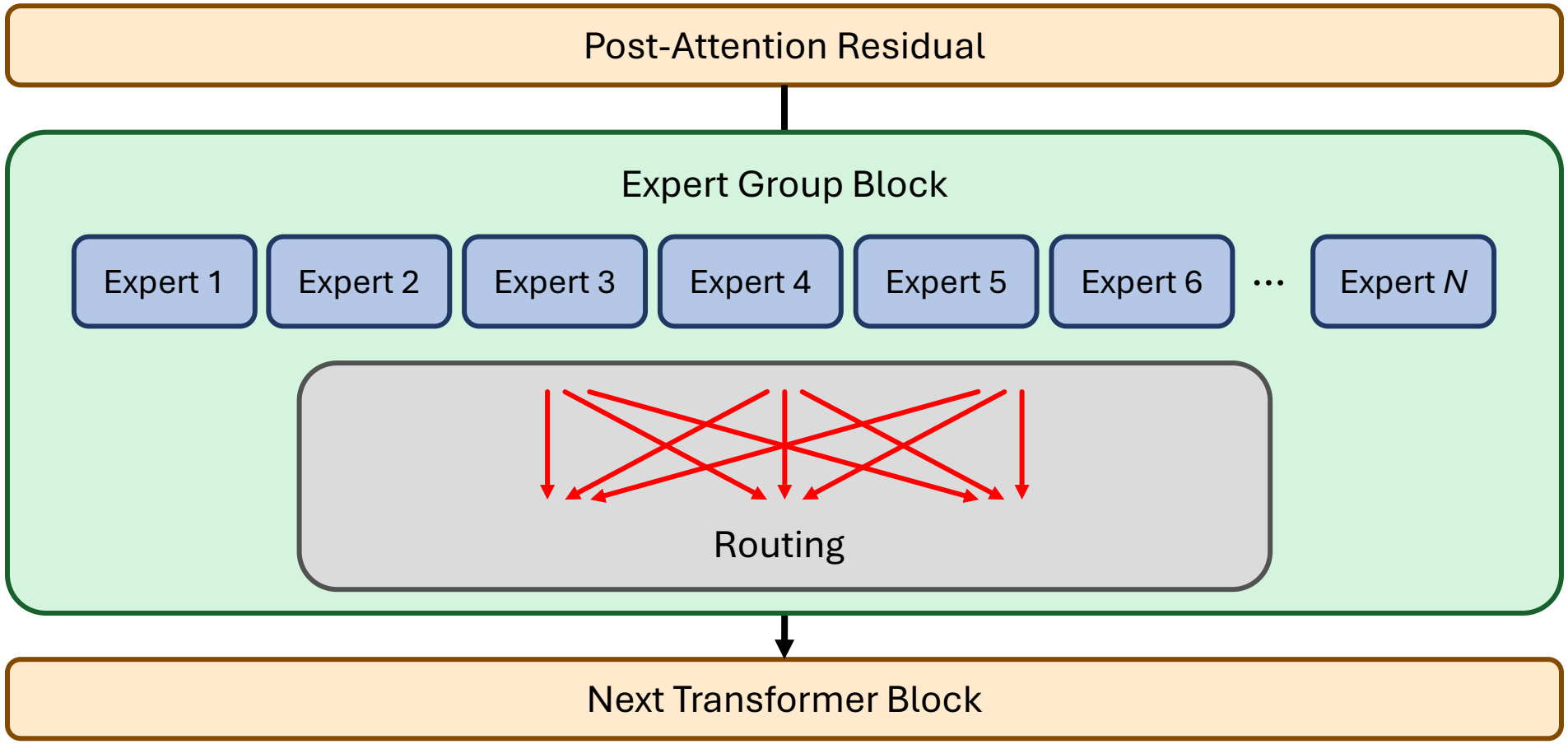


Federation of Experts Timing

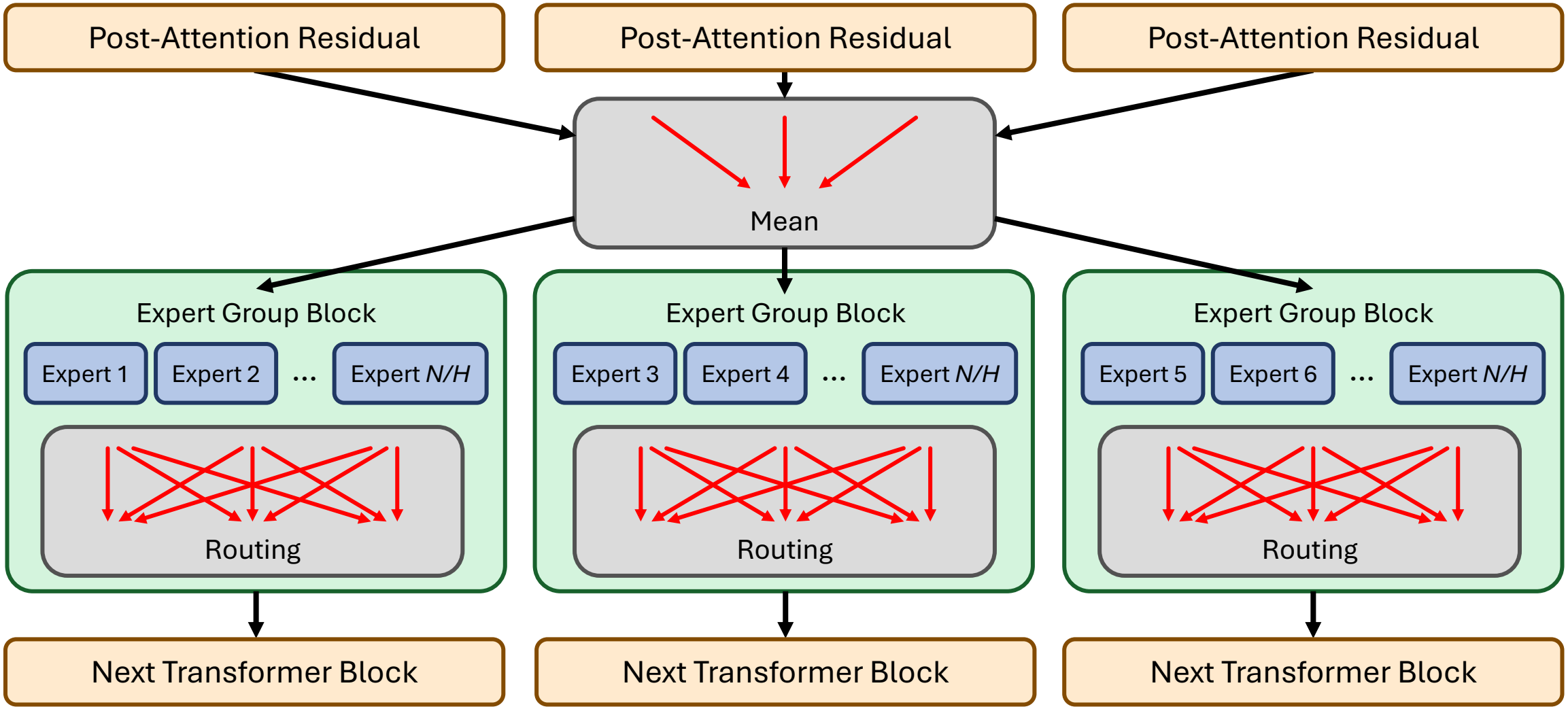
Federation of Experts

- Restructures the attention and FFN layers into $H = n_{kv}$ independent MoE groups
- Each group consists of:
  - Independent attention layer $1/H$ of the KV heads
  - Expert group with $1/H$ of the total experts
- A token's experts include k/H experts from each group
  - Tokens are routed between experts in a group, but never between groups
- Results are synchronized between groups using a sum
- Single-node: all-to-all is eliminated, only all-reduce remains
- Multi-node: all-to-all intra-node, all-reduce inter-node

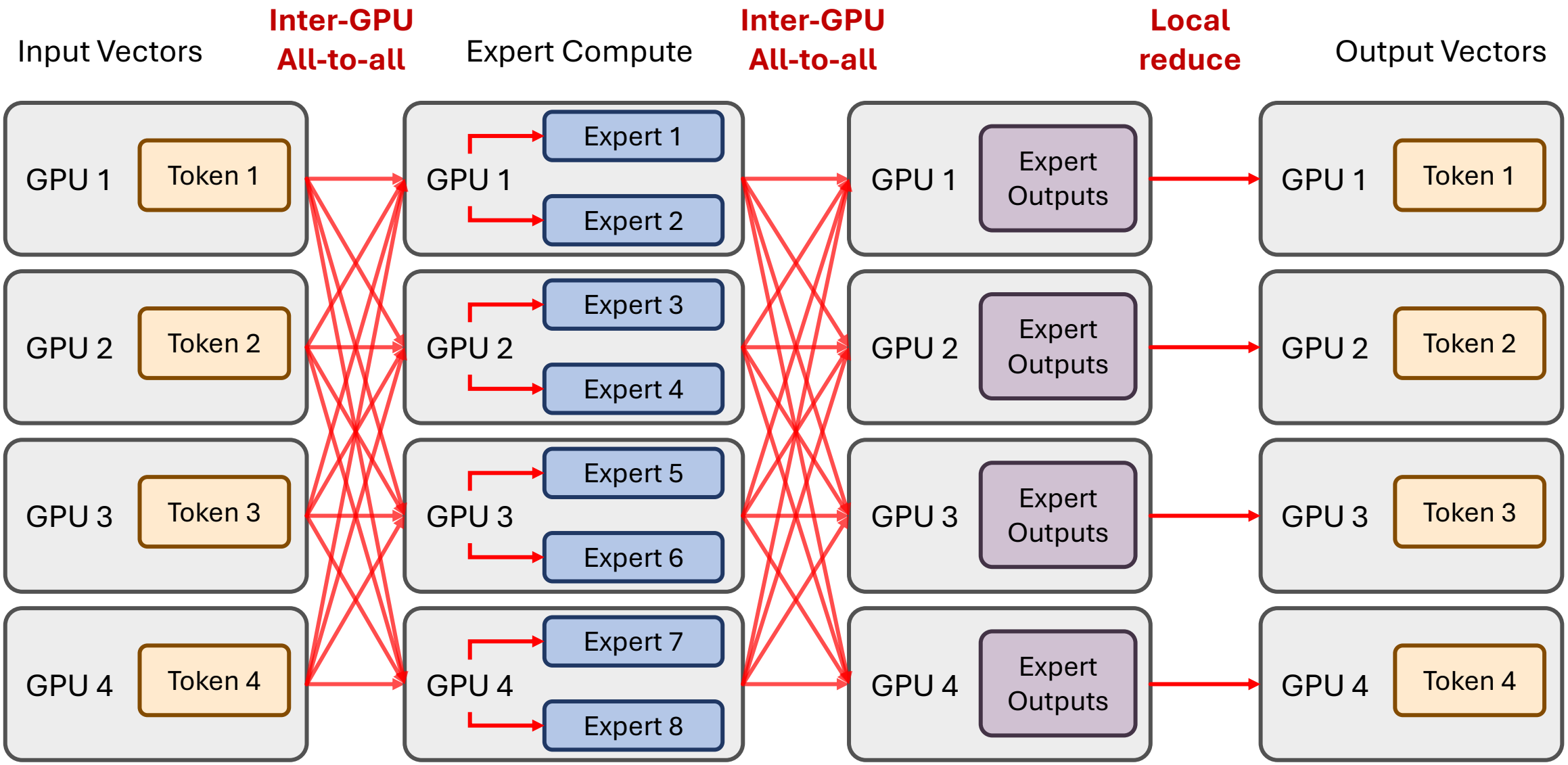
Mixture of Experts Architecture



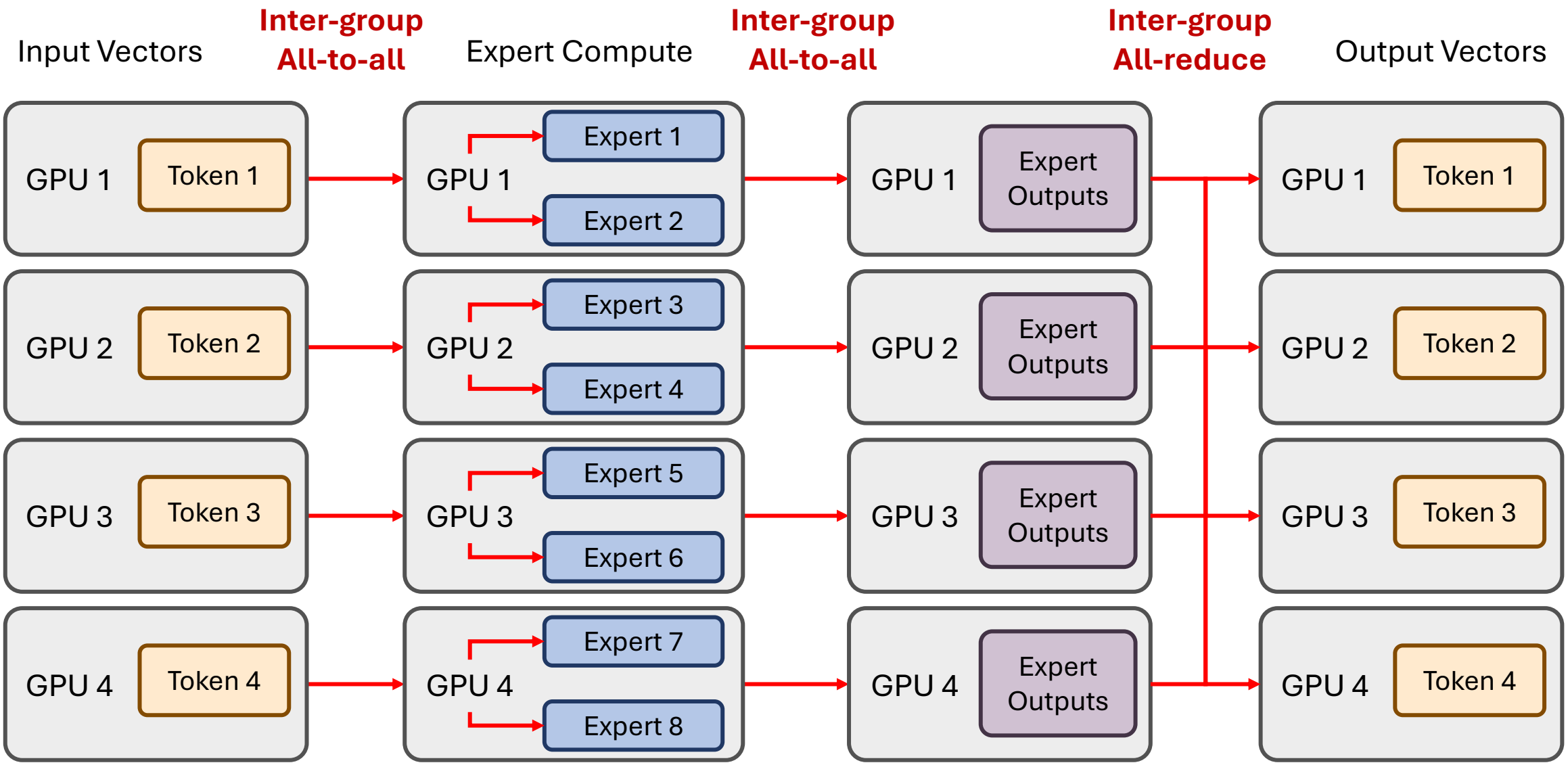
Federation of Experts Architecture



Mixture of Experts Communication



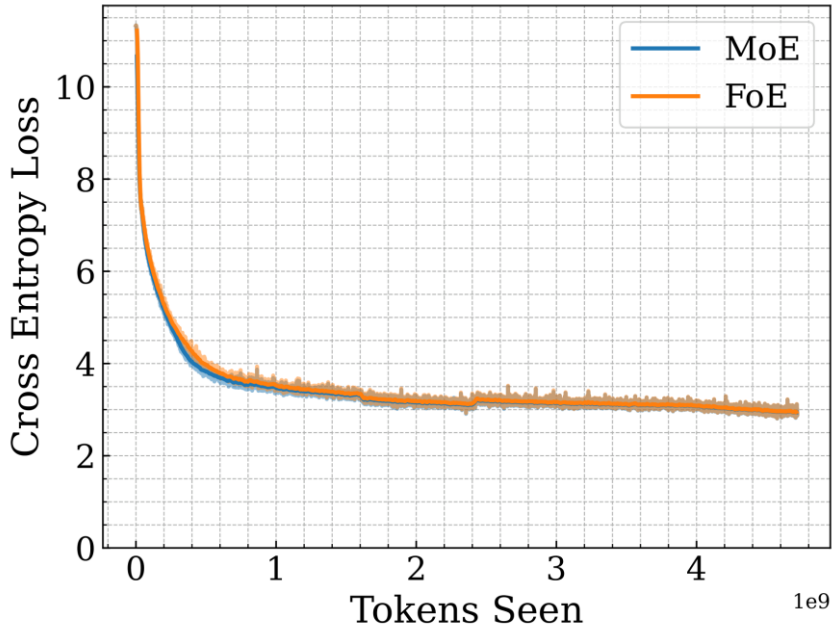
Federation of Experts Communication

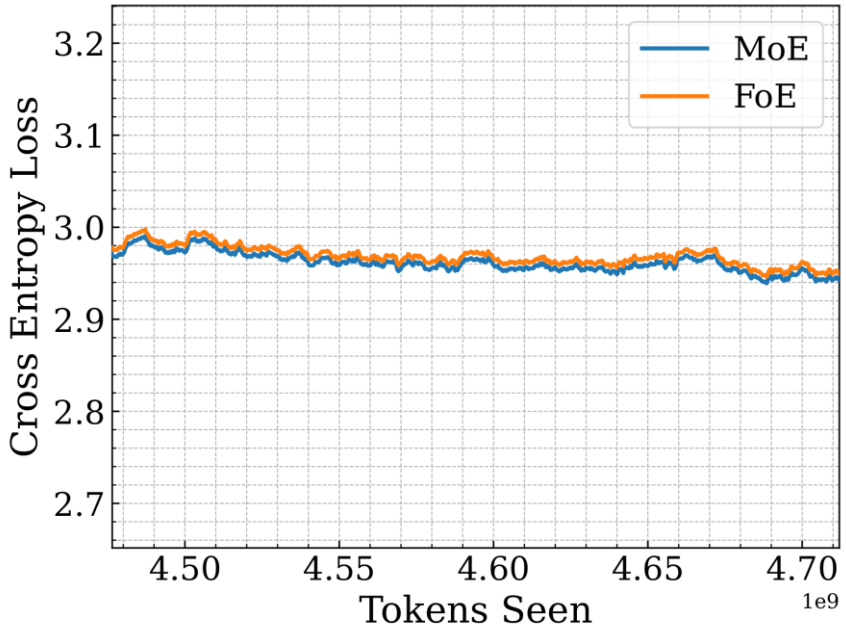


Experimental Setup

- *Hardware*: 8x H100 nodes, 80GB VRAM per GPU, SXM + Infiniband.
- *Training*: We train MoE and FoE on our own training engine built from the ground up based on TorchTitan.
- *Inference*: We implement both MoE and FoE on our own inference engine FlexServe built from the ground up based on vLLM & SGLang. This properly isolates architectural gains from framework-level engineering disparities.
- *Models*: We train 1B & 7B parameter models for both FoE & MoE and report end-to-end inference results for the 7B model.

Training Results for 1B Models

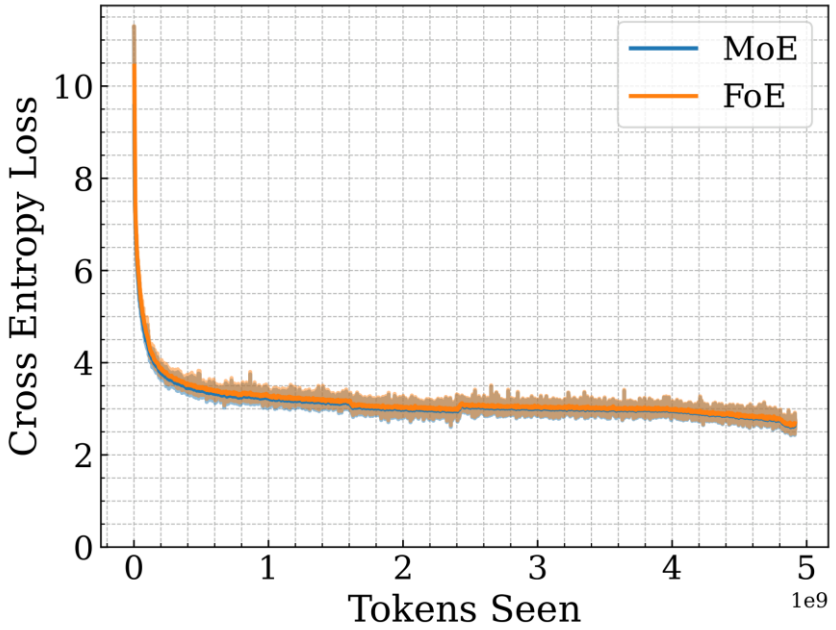


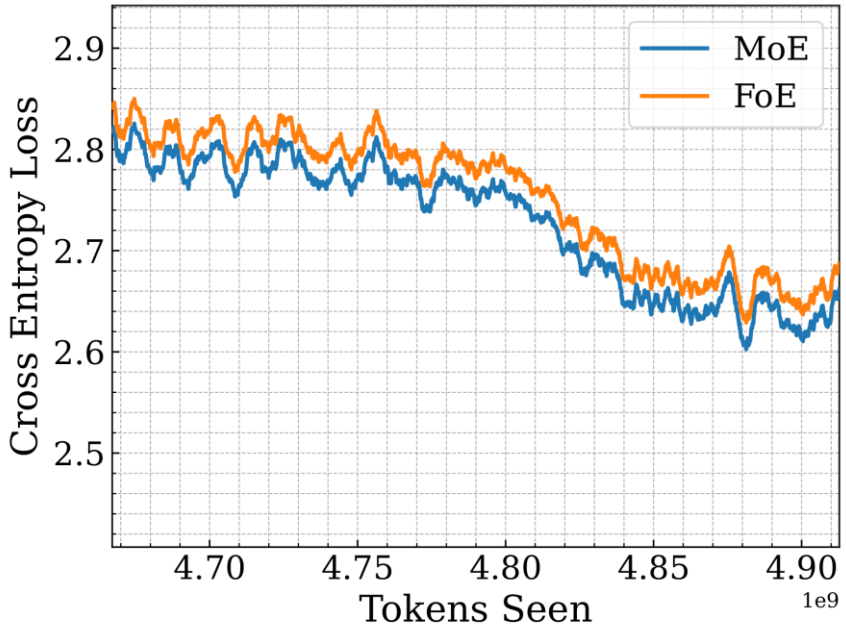


Variant	ARC-Easy	BoolQ	COPA	HellaSwag	PIQA	SciQ
MoE (1B)	53.5%	57.7%	62.0%	30.9%	63.9%	77.7%
FoE (1B)	52.7%	61.4%	61.0%	30.1%	63.8%	74.6%

At 1B total parameters, trained to ~5B tokens, FoE achieves generation comparable quality to MoE.

Training Results for 7B Models



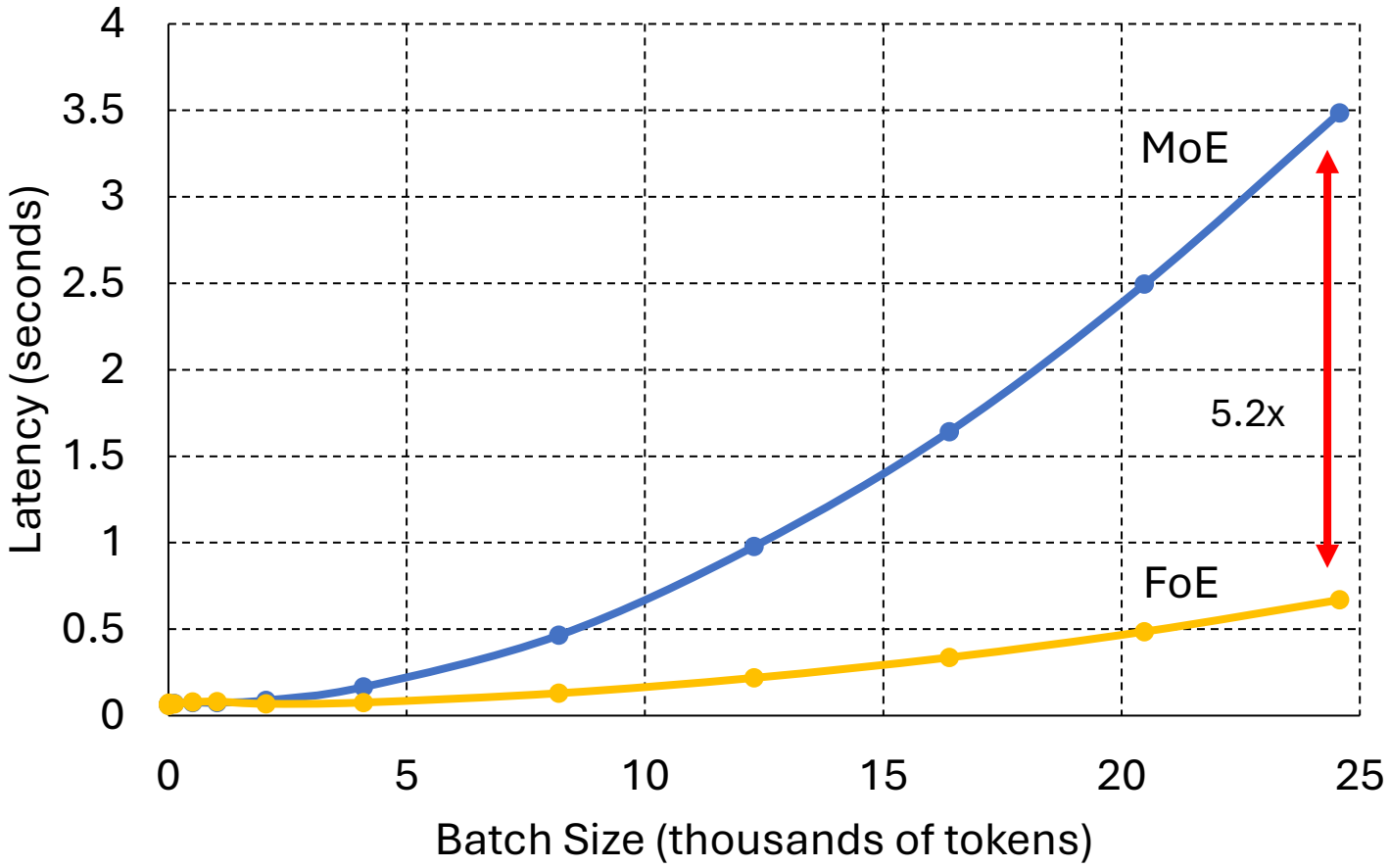


Variant	ARC-Easy	BoolQ	COPA	HellaSwag	PIQA	SciQ
MoE (7B)	59.6%	60.4%	65.0%	33.7%	66.9%	80.0%
FoE (7B)	58.2%	60.9%	66.0%	33.1%	66.6%	80.0%

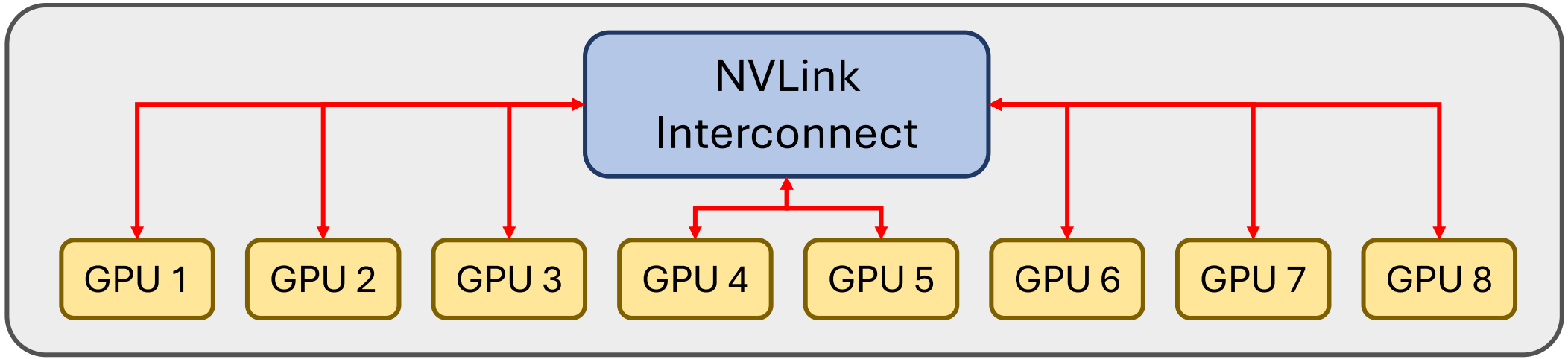
At 7B total parameters, trained to ~5B tokens, FoE achieves generation comparable quality to MoE

Forward Pass Latency

- We run synthetic forward passes at fixed batch sizes
- FoE shows dramatically improved forward-pass latency as batch size increases



Single Node Setup



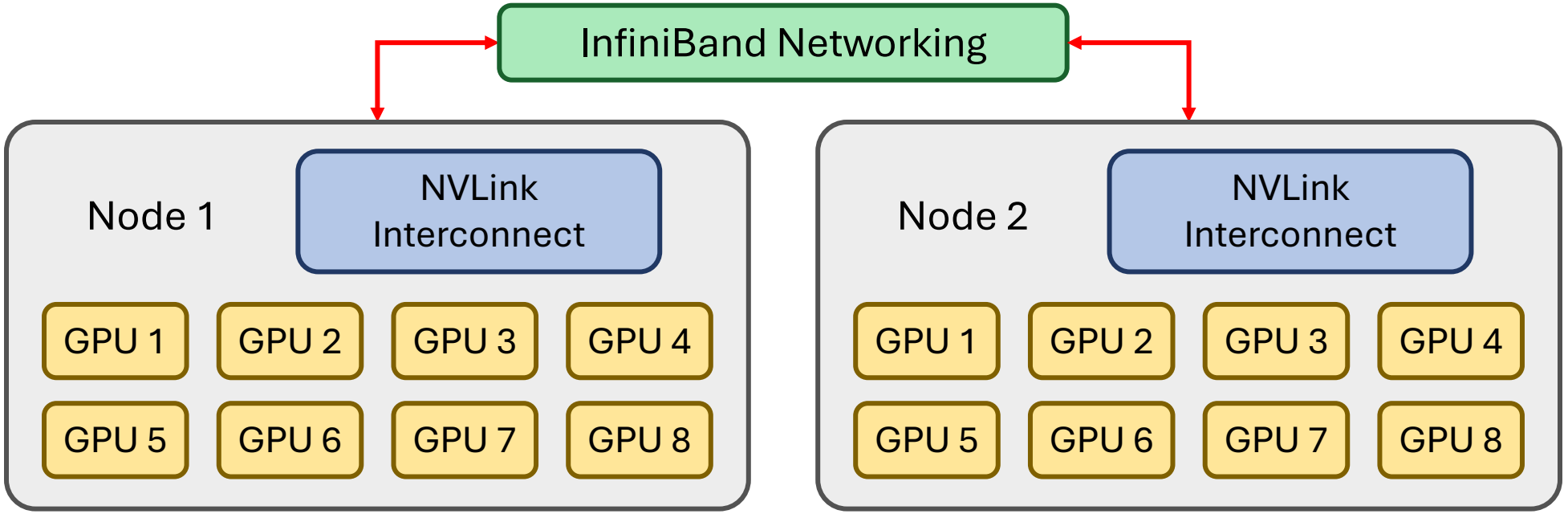
- LongBench NarrativeQA, HotpotQA, TriviaQA and PassageCount tasks (average prompt length is ~9k)
- Requests arrivals according to Poisson distribution with average inter-arrival times (scales) of 0.5, 0.75, & 1.5 seconds

Speedups on Single Node

Scale	Metric	Mean	p50	p99
0.5s	Time to First Token	*2.66x*	*2.99x*	*1.82x*
	Time Between Tokens	*1.38x*	*1.31x*	*1.53x*
	End to End Time	*1.39x*	*1.32x*	*1.54x*
0.75s	Time to First Token	*3.11x*	*3.21x*	*2.77x*
	Time Between Tokens	*1.83x*	*1.90x*	*2.06x*
	End to End Time	*1.85x*	*1.91x*	*2.07x*
1.5s	Time to First Token	*2.87x*	*2.85x*	*3.15x*
	Time Between Tokens	*1.11x*	*1.08x*	*1.53x*
	End to End Time	*1.13x*	*1.09x*	*1.56x*

Reported as ratios of MoE time vs. FoE time

Multi Node Setup



- 2 nodes with identical GPU configurations
- LongBench tasks (average prompt length is ~9k)
- Requests arrivals according to Poisson distribution with average inter-arrival times (scales) of 0.5 & 0.75 seconds

Speedups on Multi Node

Scale	Metric	Mean	p50	p99
0.5s	Time to First Token	*3.44x*	*3.57x*	*2.74x*
	Time Between Tokens	*1.36x*	*1.33x*	*1.57x*
	End to End Time	*1.37x*	*1.34x*	*1.57x*
0.75s	Time to First Token	*3.62x*	*3.59x*	*3.70x*
	Time Between Tokens	*1.95x*	*2.01x*	*2.26x*
	End to End Time	*1.96x*	*2.02x*	*2.27x*

Reported as ratios of MoE time vs. FoE time

Limitations

- Benefits only apply to distributed deployments.
  - On a single GPU, there is no all-to-all communication to reduce, thus the cross-group sums and separate attention layers become additional overhead without offsetting savings.
- Training speed is not improved.
  - Each expert group has its own residual stream, resulting in a significant backward pass overhead.

Conclusions

- FoE achieves a breakthrough in MoE inference performance by restructuring the core architecture.
  - FoE divides experts, attention, and routing into groups thus enabling memory and compute to be co-located.
  - This results in reduced communication and improved load-balancing.
- Communication does not always equal model quality.
  - Existing architectures should undergo ablation experiments to understand whether any expensive data movement is truly needed for good generation.
- Next Steps:
  - Validate model quality scaling to larger parameter counts.
  - Can an existing trained MoE model checkpoint be adapted to an FoE model?

Questions