

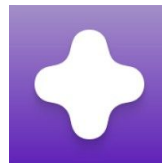
Towards Real Time Transformer Inference

18th Scheduling for large-scale systems workshop

Loris Marchal - CNRS Research Director

Gaël Delalleau - Kog CEO

Félix Wirth - (soon) PhD Student @ Kog/CNRS



Generative AI use cases evolution

We are moving from:

- Chatbots
- Classifiers

To:

- Reasoning Models
- Agentic AI



The user perspective - Chatbot

Hello assistant, what should I wear in Montréal today ?

Good morning! Here's your weather update and outfit suggestion for Montréal today, Wednesday, July 9,



The user perspective - Agentic AI

Hello assistant, what should I wear in Montréal today ?

Searching the web







The user perspective - Agentic AI

Currently 71° · Partly sunny

Montreal, QC, Canada



	8 am	65°	Partly sunny
	9 am	67°	Mostly sunny
	10 am	70°	Mostly sunny
	11 am	73°	Mostly sunny
	12 pm	76°	Mostly sunny
	1 pm	79°	Mostly sunny
	2 pm	79°	Mostly sunny
	3 pm	80°	Mostly sunny

Show more



The user perspective - Agentic + Reasoning

Hello assistant, what should I wear in Montréal today ?

Sure! I'll look up today's weather forecast in Montréal to suggest what you should wear, including considerations for temperature, precipitation, and time of day.
I'll be back shortly with personalized recommendations.

Starting research



The user perspective - Agentic + Reasoning

Research completed in 7m · 16 sources · 36 searches

 What to Wear in Montréal Today



Weather Forecast and Clothing Recommendations for Montréal on July 9, 2025

Weather Forecast Details for July 9, 2025

Montreal is expected to have **warm and partly sunny** conditions on July 9, 2025. According to the latest forecast, the day will reach a high of about **27°C** (feeling like ~32°C with humidity) weather.gc.ca. The morning



The user perspective

Gains:

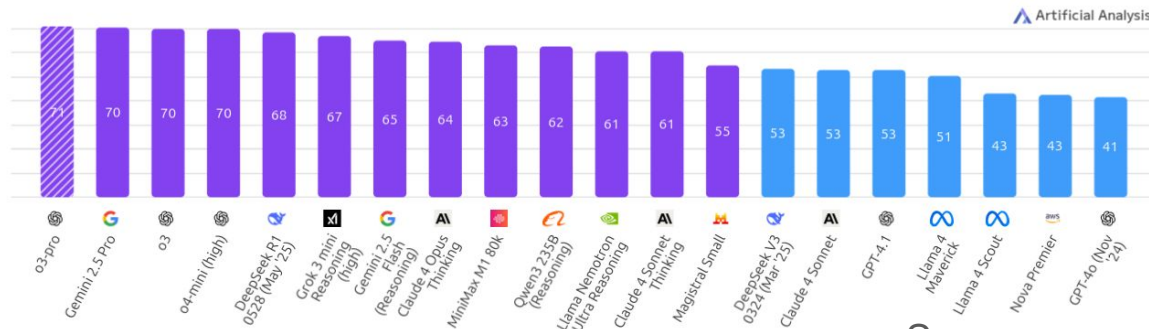
- Quality
- Relevance
- Better understanding

Artificial Analysis Intelligence Index by Model Type

Artificial Analysis Intelligence Index incorporates 7 evaluations: MMLU-Pro, GPQA Diamond, Humanity's Last Exam, LiveCodeBench, SciCode, AIME, MATH-500

Estimate (independent evaluation forthcoming)

Reasoning Model Non-Reasoning Model



Source:

[Artificial Analysis](#)



The user perspective

Gains:

- Quality
- Relevance
- Better understanding

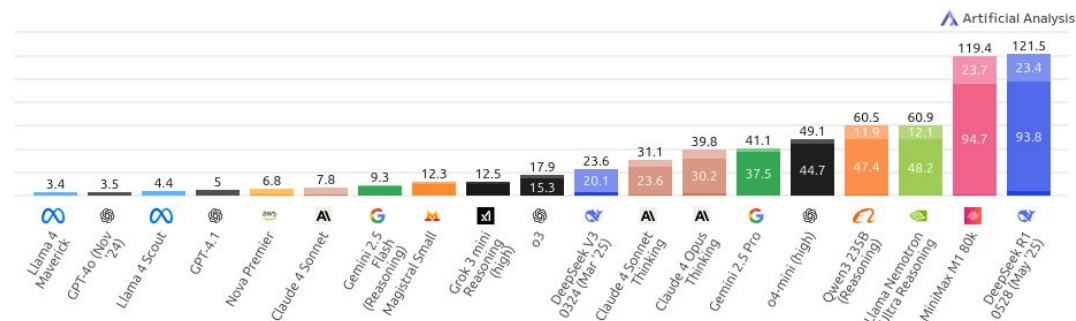
Loss:

- Waiting time

End-to-End Response Time

Seconds to Output 500 Tokens, including reasoning model 'thinking' time; Lower is better

■ Input processing time ■ 'Thinking' time (reasoning models) ■ Outputting time



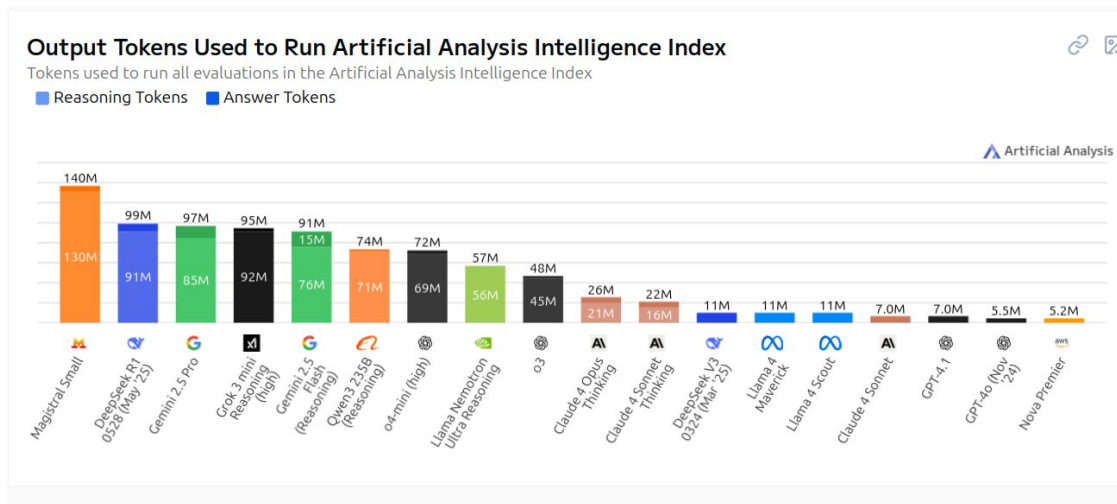
Source:

[Artificial Analysis](#)



Why the waiting time ?

- Obtaining the final answer requires generating ‘intermediary results’: Reasoning traces, calling tools / browsing web, etc.
- Pipelining reasoning calls make it worse



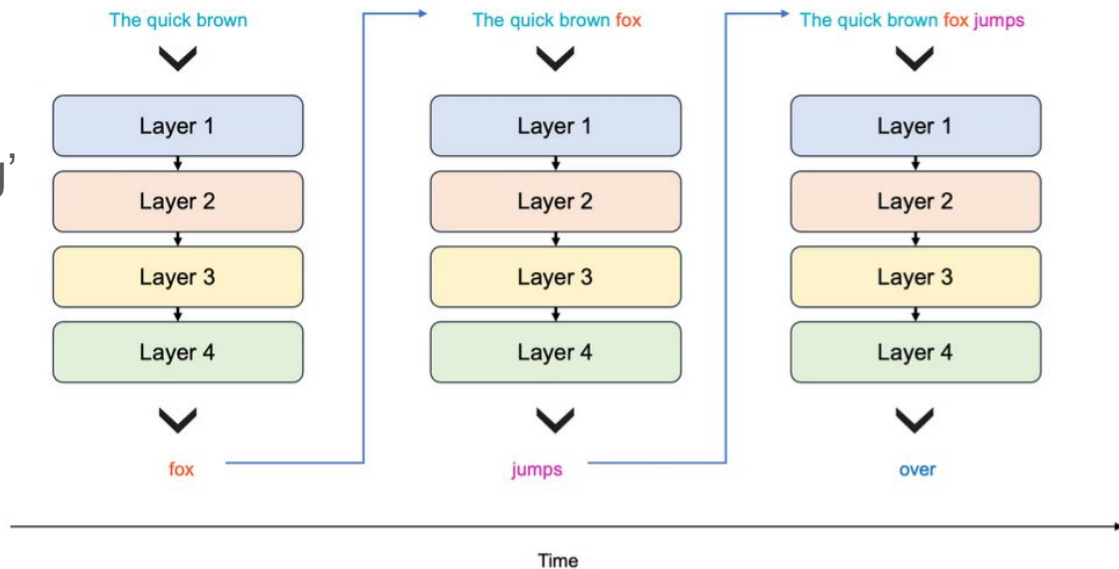
Source:
[Artificial Analysis](#)



Autoregressive Generation

- The whole model is used for every new token
- First pass is named 'Prefilling'
- Every following pass is 'Decoding'

We focus on Decoding today !

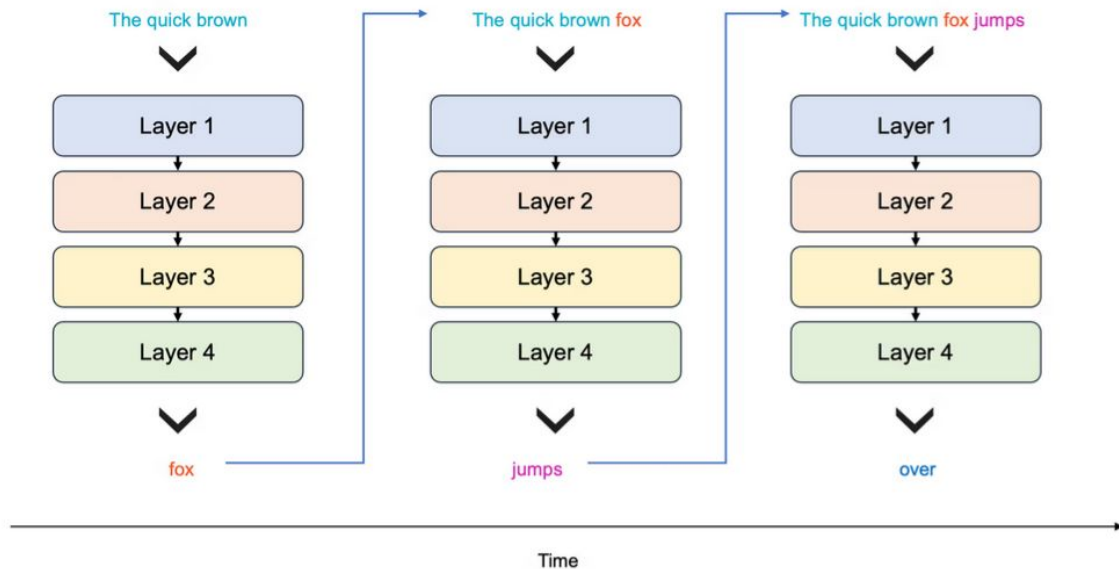


Source: [Cerebras](https://cerebras.ai/)



Autoregressive Generation

- Each pass is memory bound
- This is usually alleviated by predicting multiple sentences at a time (higher batch size)



How to optimize per user speed ?

We focus on batch size 1

Source: [Cerebras](#)



Efficient Inference Solutions

Model compression:

- Pruning
- Quantization

Smarter decoding algorithm

- Speculative decoding
- KV Cache

Leveraging hardware

- Kernel optimization
- Parallelization



Efficient Inference Solutions

Model compression:

- Pruning
- Quantization

Smarter decoding algorithm

- Speculative decoding
- KV Cache

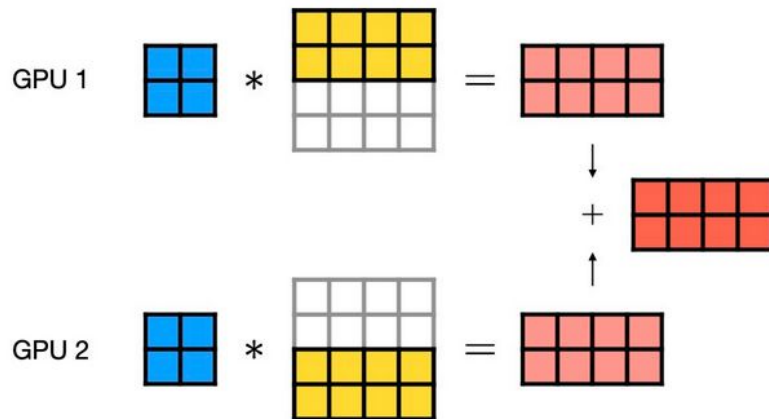
Leveraging hardware

- Kernel optimization
- Parallelization



Parallelization for speed

- Reduce memory reads per GPU
- Split the model along weights
- Combine Column and Row wise splits to reduce communications required
- Almost divides memory read by number of GPU used

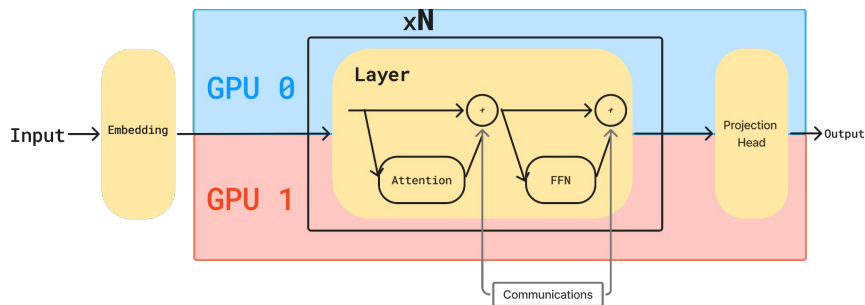


Source: [Lightning AI](#)



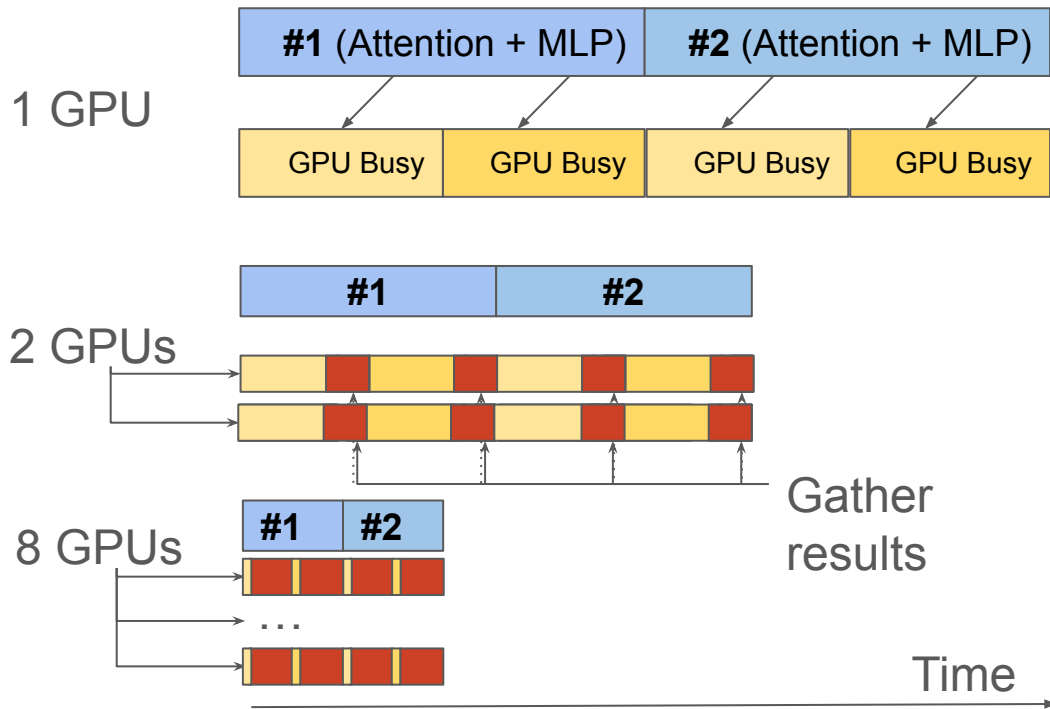
Application and Limitations to Transformer

- [Megatron-LM](#) efficiently applies this at scale to every layer - Using 2 communications per layer
- Poor performance scaling is observed at small batch



Multi GPU Limitations

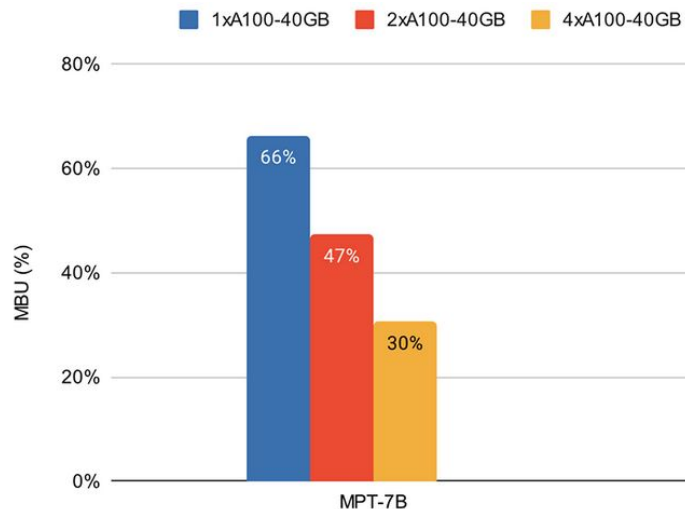
- Launching the comm takes ~few μ s
- In this span of time, each GPU can read multiple MBs of data
- On small models and batch sizes, it dominates the runtime.



Multi GPU Limitations

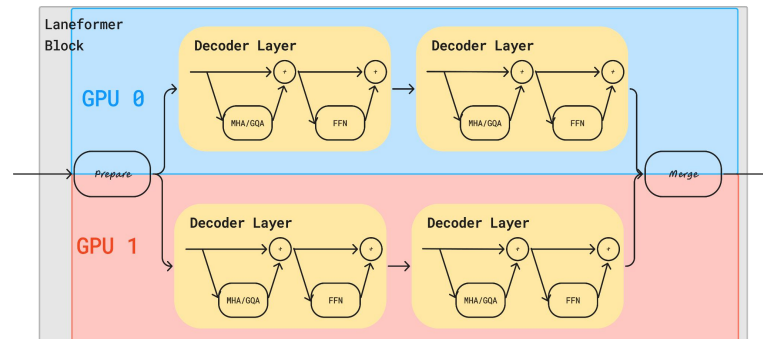
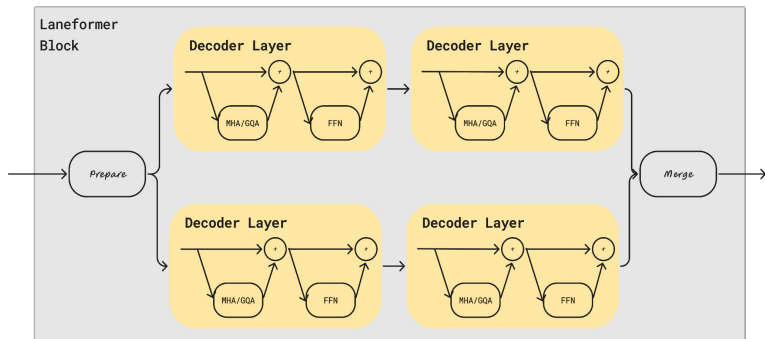
Benchmarking from [Databricks](#) showing the reduced memory usage:

More GPU, less efficiency!



Solution : Natively Parallel Transformer

- We propose to break the information flow to avoid immediate data need across workers
- Remove all but one communications !
- We call the parallel sequences of layers 'Lane'



Building the model

Given a target model size, we use hyperparameters from the [Chinchilla](#) paper: number of layers, hidden size, attention dimension...

For lane number L , we solve hidden size to keep the budget similar while keeping other hyperparameters constant: $h' = h / \sqrt{L}$.

Since the architecture is deeply changed, we need to retrain it.

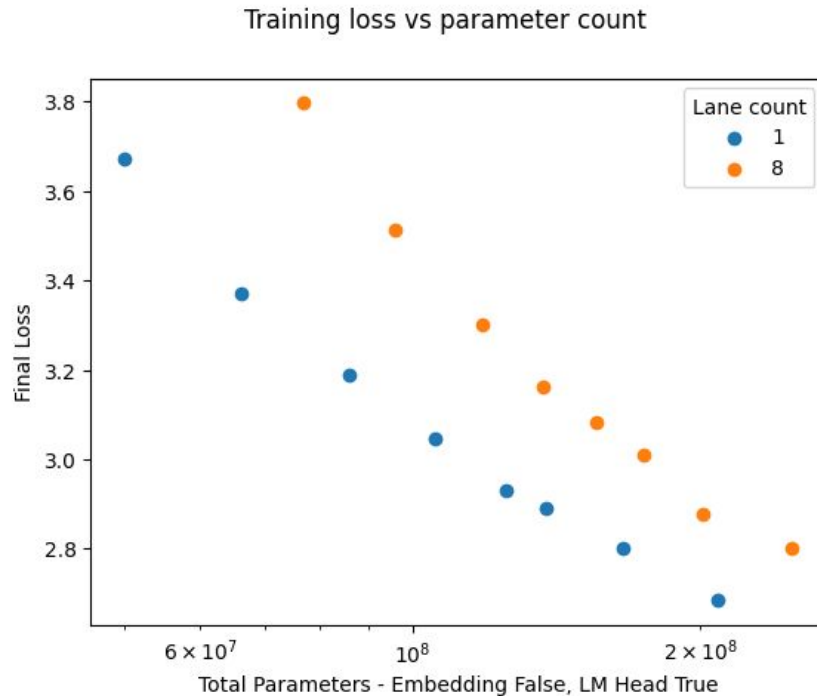


Initial Scaling Laws

Our architecture changes embedding and output parameter budget

We count 'parameters read' for inference fair comparison.

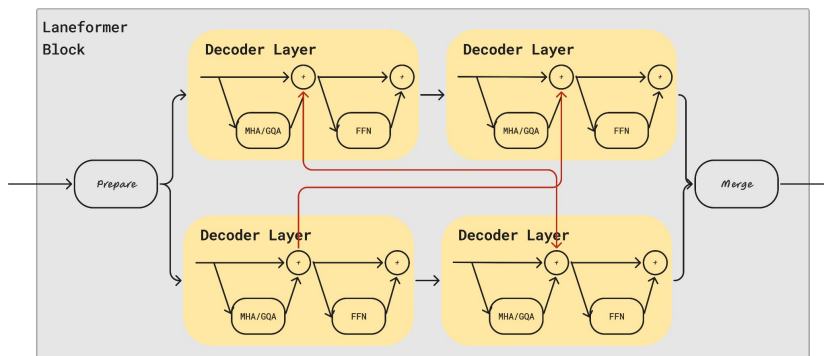
How to improve ?



Reordering information flow

We reintroduce information sharing in a delayed setting:

- Overlap the communication with other layers computations
- Size the delay according to inference setting (e.g. GPU speed)
- Aggregate using basic mean or sum with 'up to date' hidden states



Delayed communication experiments

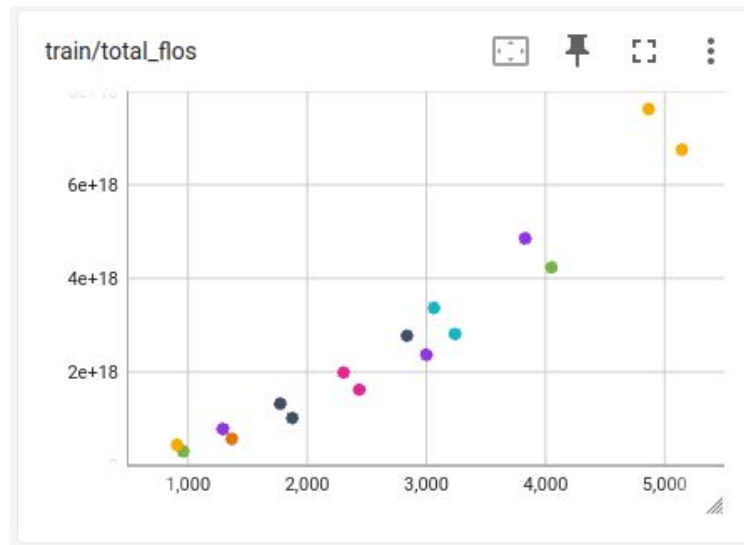
- Current results show lower performance than no communication
- We are investigating impact on the gradients
- Parametrized combination instead of mean/sum
- Avoid changing the whole hidden states?
- Different starting/ending module
- Ongoing !



From Pre to Post-training

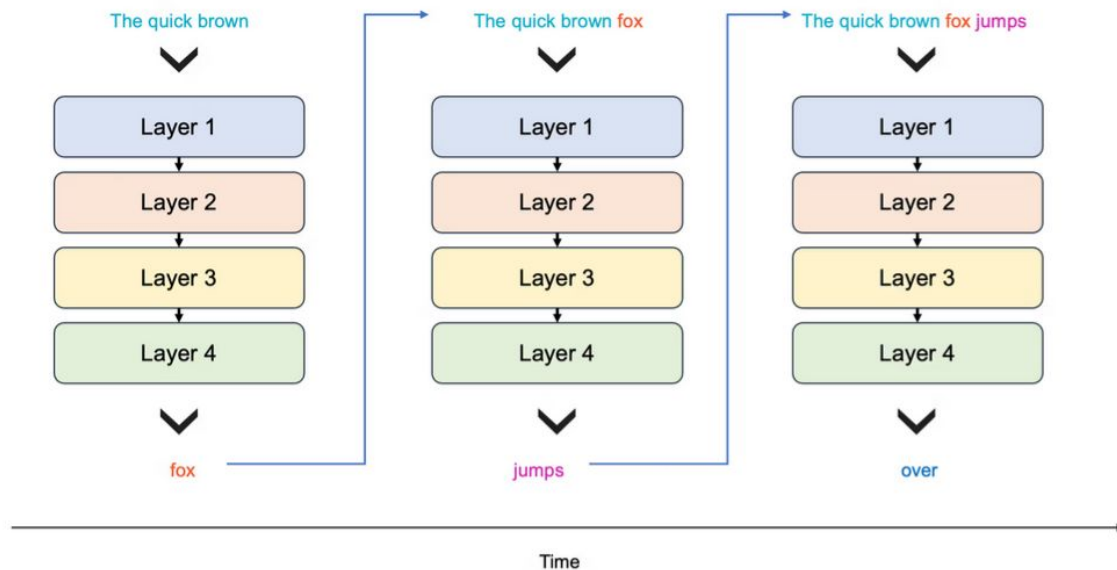
Our splitting increases the already high compute budget for training model.

How to leverage post training to avoid paying the pre-training toll ?



AI bottleneck : Long Context vs KV Cache

Inference is speeded up by caching redundant results (KV Cache), growing with longer context. Reading the cache can easily dominate the inference speed.



Source: [Cerebras](https://www.cerebras.ai/)



AI bottleneck : Long Context vs KV Cache

Inference is speeded up by caching redundant results (KV Cache), growing with longer context. Reading the cache can easily dominate the inference speed.

KV Cache is usually reduced by using:

- Hybrid architecture (e.g. SSM)
- Sliding Window Attention

Both techniques show limitations at long context sizes, clashing with modern needs.



GPU Bottleneck : Kernel Launches, internal synchronizations

- Launching a compute kernel usually takes a few microsecond, restraining the number of decoding per second
- Fusing kernels removes them
- It also removes implicit synchronizations between kernels, we need to explicitly reintroduce them: We traded launch time for inter device synchronization.



Thank you!

Towards Real Time Transformer Inference

Loris Marchal - CNRS Research Director

Gaël Delalleau - Kog CEO

Félix Wirth - (soon) PhD Student @ Kog/CNRS

