

Hello!

CS 744: NANOFLOW, VLLM

Shivaram Venkataraman

Spring 2025

ADMINISTRIVIA

Assignment grading



~80% done

↳ day or two

Assignment 2

→ grades

Project Preferences

↳ emails next day or so

early next week finalize

Inference \rightarrow serve models well! **SELF ATTENTION**

maximizing throughput \leftarrow
The first token τ between tokens

Input vectors $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t$

Output vectors $\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_t$

Attention: weighted average over all the input vectors

$$\mathbf{y}_i = \sum_j w_{ij} \mathbf{x}_j.$$

$$w'_{ij} = \mathbf{x}_i^T \mathbf{x}_j.$$

$$w_{ij} = \frac{\exp w'_{ij}}{\sum_j \exp w'_{ij}}.$$

for each output vector

ML Training
 \rightarrow Data Parallel
Pipe line
Fault Tolerance

QUERIES, KEYS, VALUES

Every input vector \mathbf{x}_i is used in three different ways

- Query: Compared to every other vector for its own output \mathbf{y}_i
- Key: Compared to every other vector for the output vector \mathbf{y}_j
- Value: Part of the weighted sum to compute each output vector

$$\underline{\mathbf{q}}_i = \mathbf{W}_q \mathbf{x}_i$$

$$\underline{\mathbf{k}}_i = \mathbf{W}_k \mathbf{x}_i$$

$$\underline{\mathbf{v}}_i = \mathbf{W}_v \mathbf{x}_i$$

every input vector

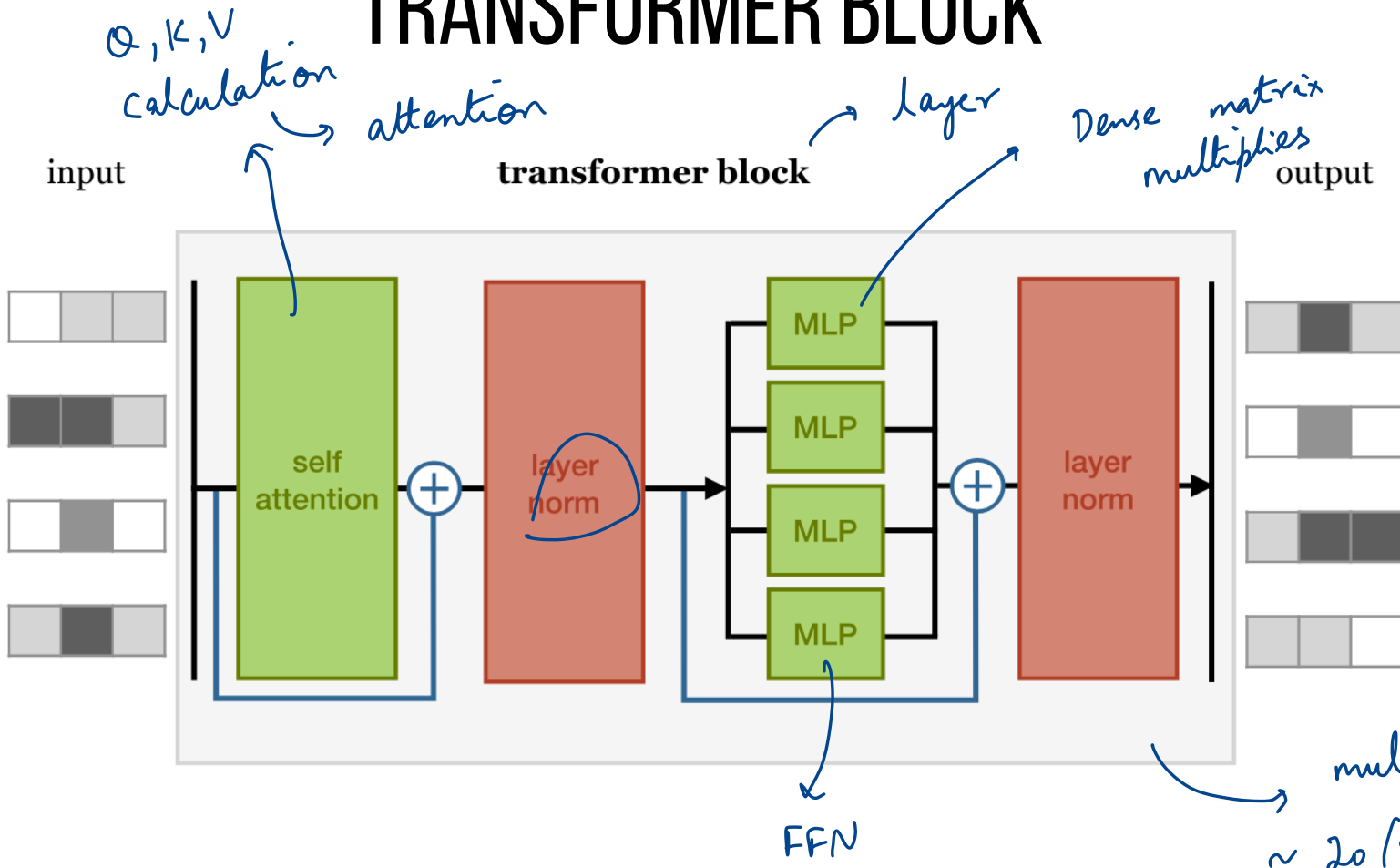
multiplying weight matrix

output vectors ←

in attention

$$w'_{ij} = \mathbf{q}_i^T \mathbf{k}_j$$
$$w_{ij} = \text{softmax}(w'_{ij})$$
$$\mathbf{y}_i = \sum_j w_{ij} \mathbf{v}_j$$

TRANSFORMER BLOCK



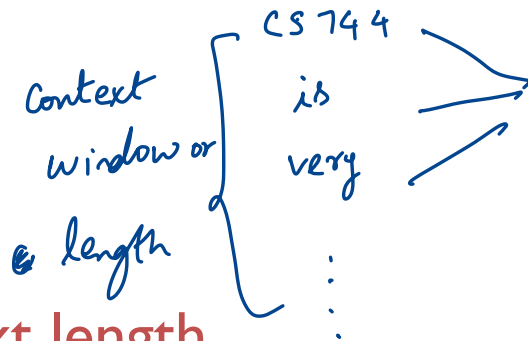


KV CACHE, PREFILL VS DECODE

$$q_i = W_q x_i$$

$$k_i = W_k x_i$$

$$v_i = W_v x_i$$



Cache Key and Value vectors for the **context length**

Minimize re-generation of key and value vectors for prior tokens.

↳ repeatedly evaluate k_i, v_i for all prior tokens

Pre-fill: Digests all the input tokens (prompt). Fill KV Cache

Decode: Generate token-by-token till termination

Prefill

- Input all the tokens at once
- diff resources bottleneck
- get all of the prompt tokens

Decode

- single token at once
- based on what was generated previously

Prompt

"CS744 is very"

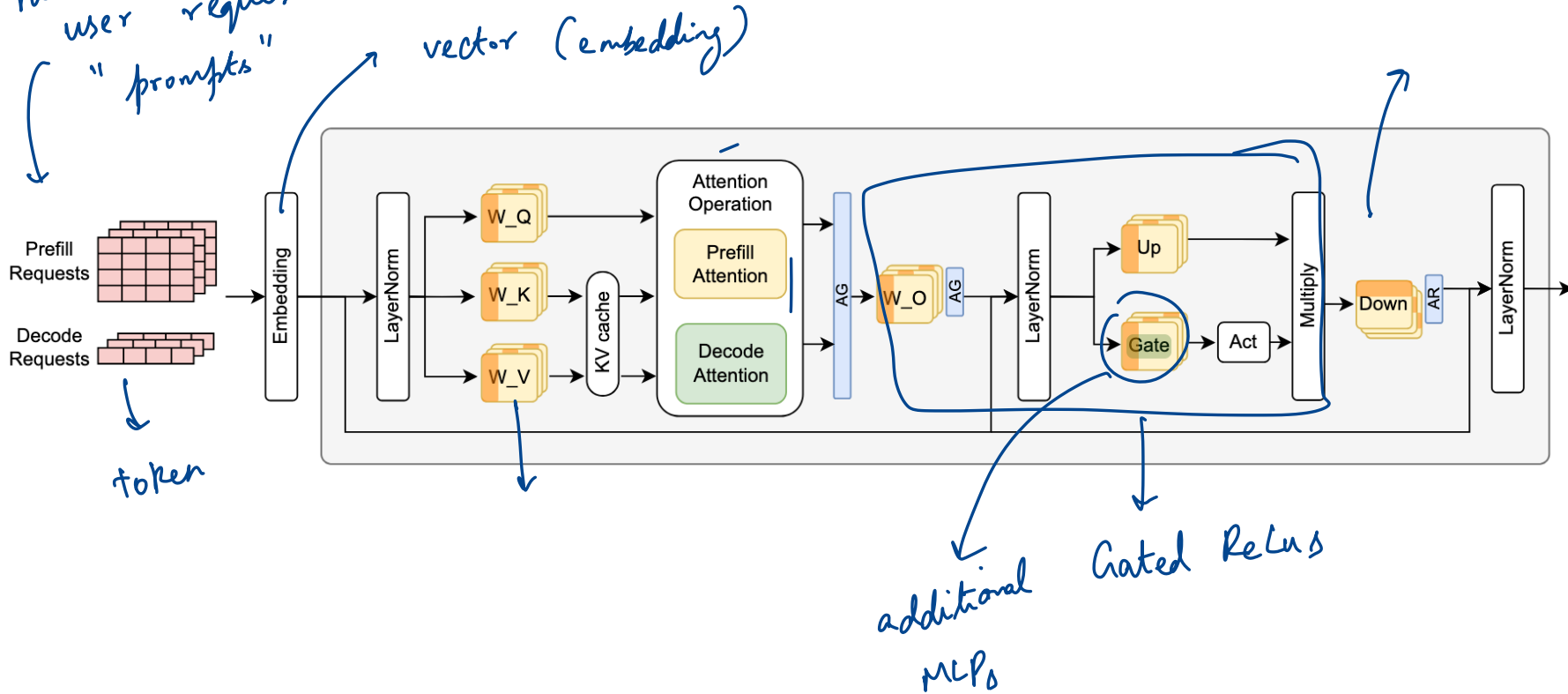
Next token / prompt generation

→ "interesting class about Big Data"

PUTTING IT ALL TOGETHER

number of user requests
"prompts"

Llama-3



HOW TO ENSURE HIGH THROUGHPUT?

Approach: Try to derive hardware bounds.

Investigate which hardware resource is the bottleneck

How many
FLOPs can
you do
for 1 byte
of data
read

GPU Model	Release Year	NetBW (GB/s)	Compute (FP16 GFLOP/s)	MemBW (GB/s)	Ratio (FLOP/B)
V100	2017	300	125,000	900	139
A100 - 40G	2020	600	312,000	1555	200
A100 - 80G	2021	600	312,000	2000	156
H100	2023	600	989,000	3352	295
H200	2024	900	989,000	4800	206
B100	2024	1800	1,800,000	8000	225
B200	2024	1800	2,250,000	8000	281
MI250	2021	800	362,000	3352	107
MI300	2023	1024	1,307,000	5300	246

ratio is
~ 200 - 280 ...

PROFILING RESULTS

Perf model captures trends

by performance model

multiplying input with w_k, w_q, w_v

Operation	Compute (GFLOP)	Memory Movement (GB)	Network Usage (GB)	$T_{compute}$ (ms)	T_{mem} (ms)	T_{net} (ms)	Measured Time (ms)
GEMM-KQV	27487.8	19.5	0	11.01	1.22	0	16.08
GEMM-O	21990.2	16.1	0	8.81	1.01	0	16.01
GEMM-UG	153931.6	96.6	0	61.67	6.04	0	69.92
GEMM-D	76965.8	49.7	0	30.84	3.11	0	34.96
Decode Attention	3665.9	462.2	0	1.47	28.89	0	35.60
Prefill Attention	916.3	2.1	0	0.37	0.13	0	4.56
Communication	18.8	75.2	75.2	0.01	4.70	31.33	47.92
Total				114.17	45.09	31.33	

MLP layers after attention

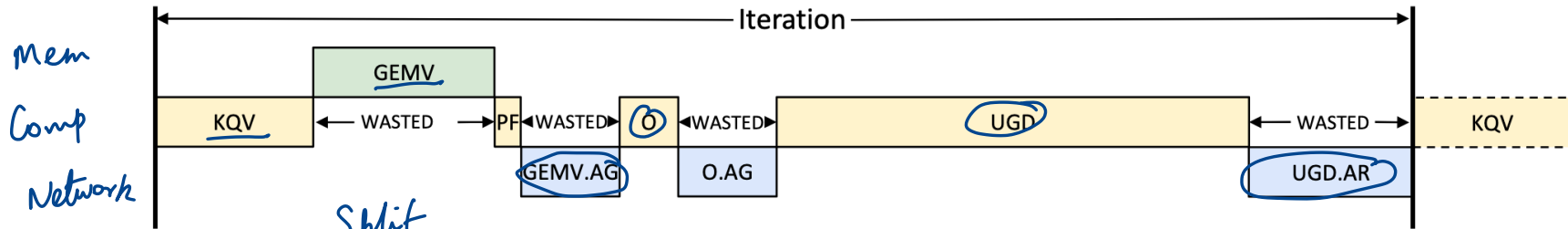
Compute is bottleneck

mem bw is bottleneck

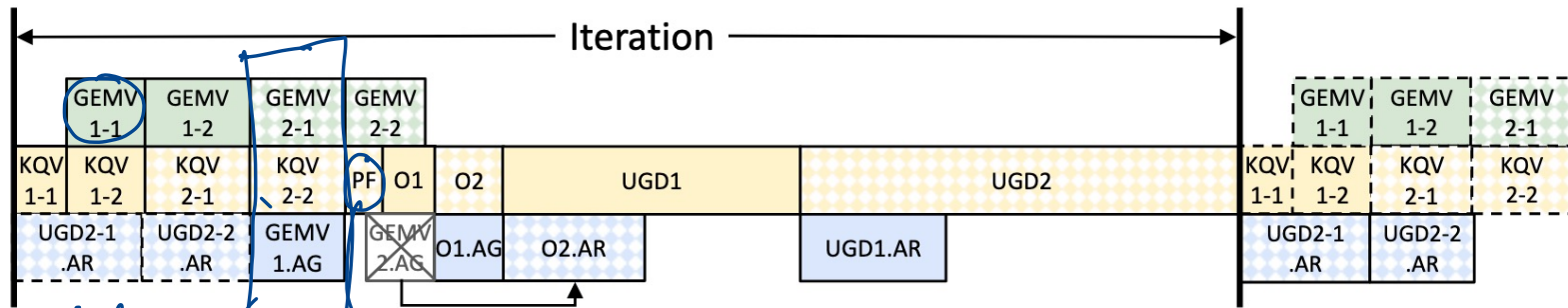
overall bottleneck

HOW TO IMPROVE UTILIZATION? NANO BATCHES

Form 1 batch



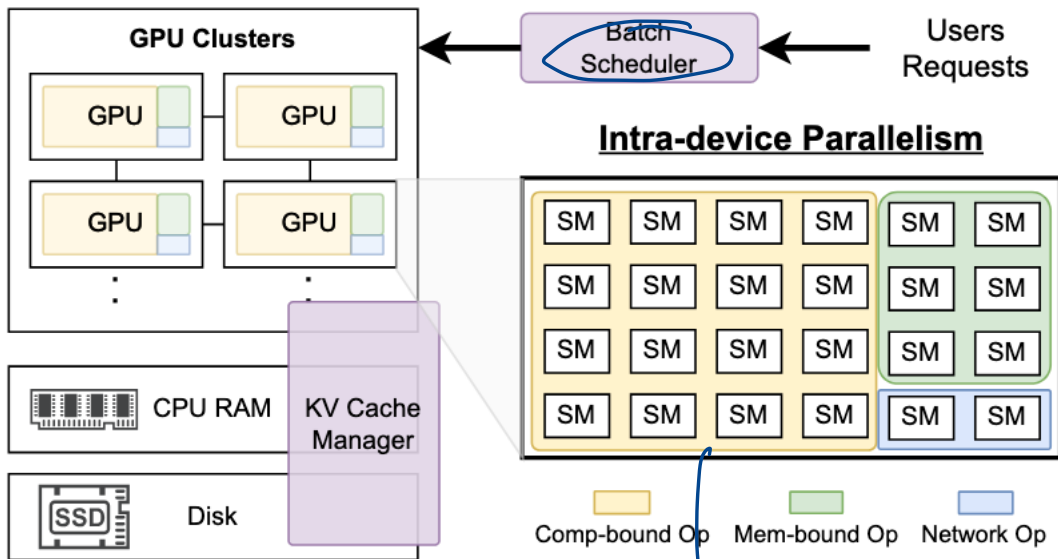
1 batch \rightarrow Split nano batches \rightarrow Work on each nano batch is independent



at the same time \leftarrow 1 nano batch

DESIGN

2049 → +put reduces



Pick batch sizes which maximize throughput (2048)

Minimize number of nanobatches, avoiding pipeline bubbles

→ too small
→ lose tput

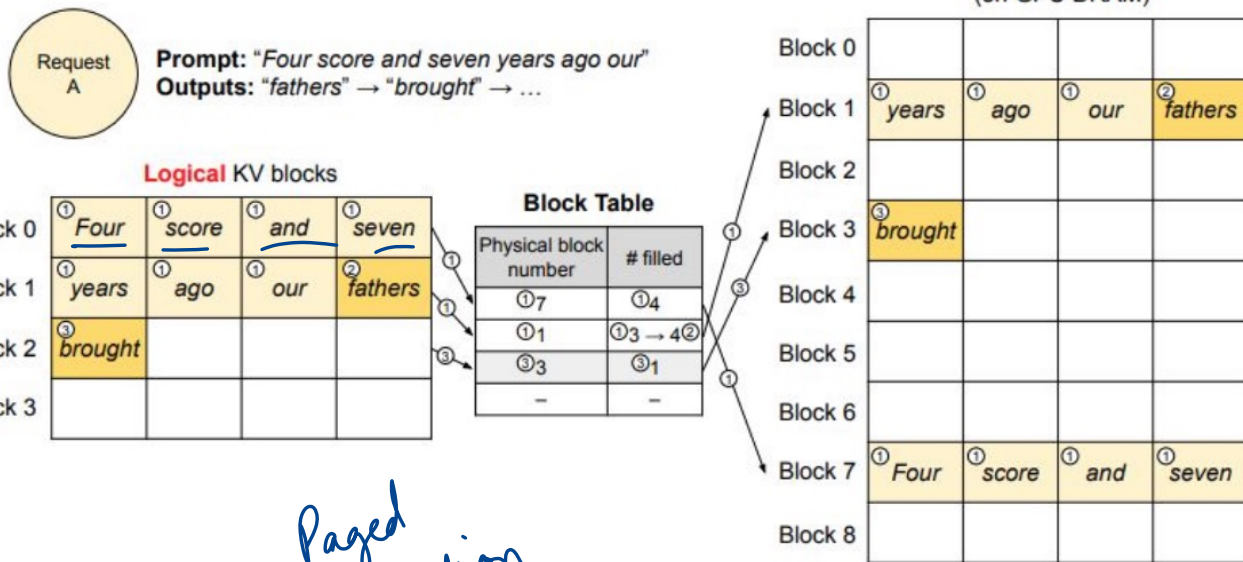
Intra-device: Search for number of SMs to allocate for each nanobatch

Streaming Multi Processors

DESIGN: KV CACHE MANAGEMENT

Offload unused KV caches to SSDs

Load KV cache back for a multi-round use case



Paged Attention

fragmentation /
memory waste

some of them
be on other GPUs

SUMMARY

LLM Inference: Mix of compute, memory bound ops

Key idea: Minimize idle resources through pipelining

Careful memory management, scheduling with nanobatches



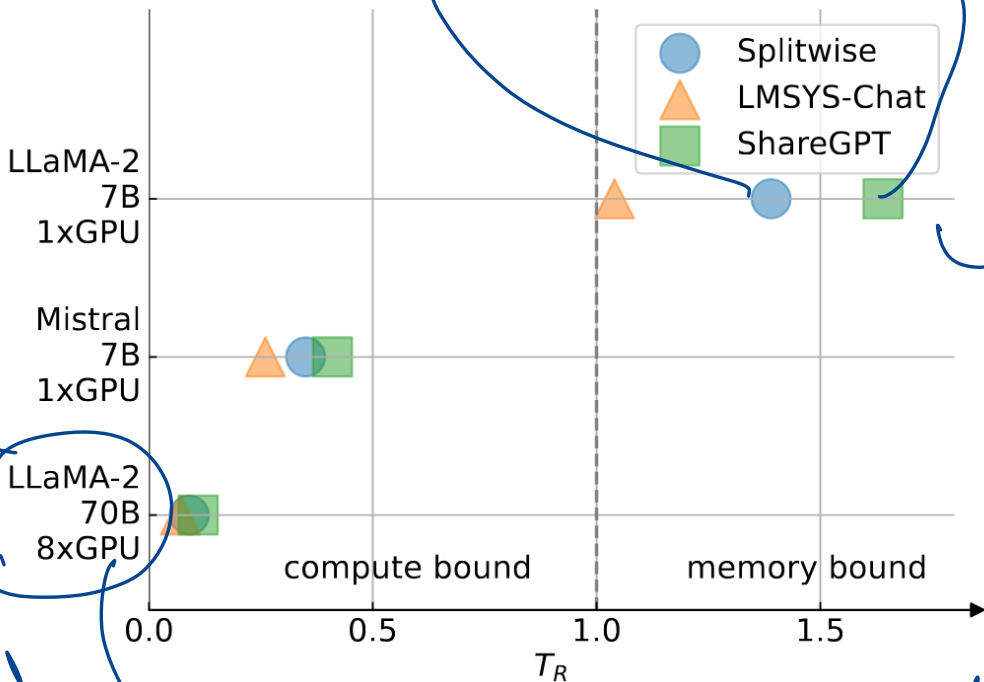
DISCUSSION

<https://forms.gle/2fnsyMI2xEWtj7TA7>

Prompt = 1155

Output is bigger (322)
prompt is smaller (246)

ratio of flops / byte



bigger prompts } → Prefill
output sizes } → decode

bigger model is more compute bound

grouped query attention

f_x more mem bw
 f_y more compute

→ model is partitioned across 8 GPUs

What are some similarities of nanoflow with prior papers we have read in the class so far? What are some differences?

Similarities

→ Constrained resource scheduling (small batches)

→ Pipelining is a good idea!

Differences

Intra device
vs

inter device

Tensor
parallelism
used in
inference

NEXT STEPS

Next class: Scheduling