Data Pre-processing Challenges in ML Pipelines

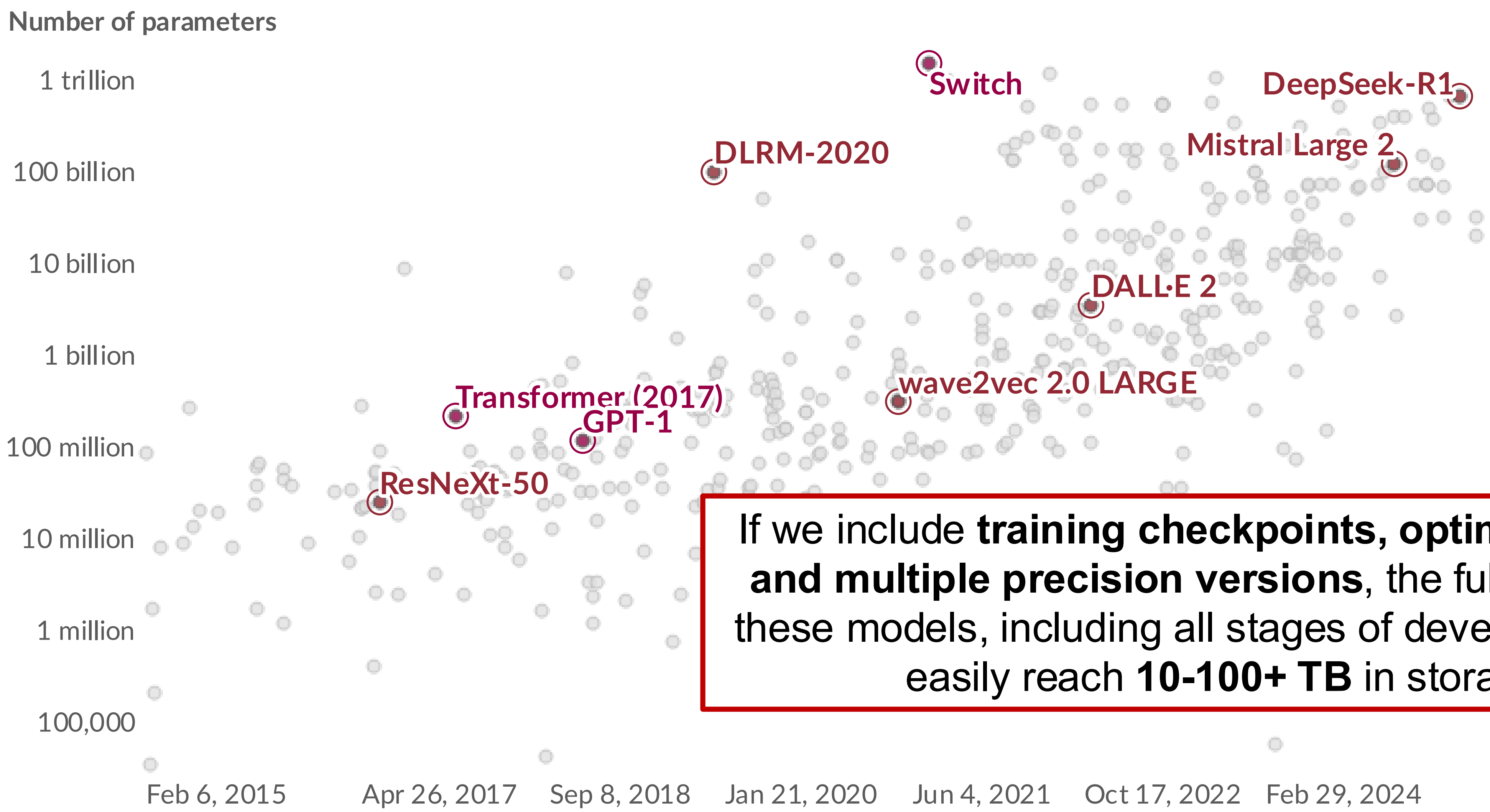
Oana Balmau, McGill University

18th Scheduling for Large-Scale Systems Workshop
ETS Montréal, Jul 8 2025

McGill

ML Commons

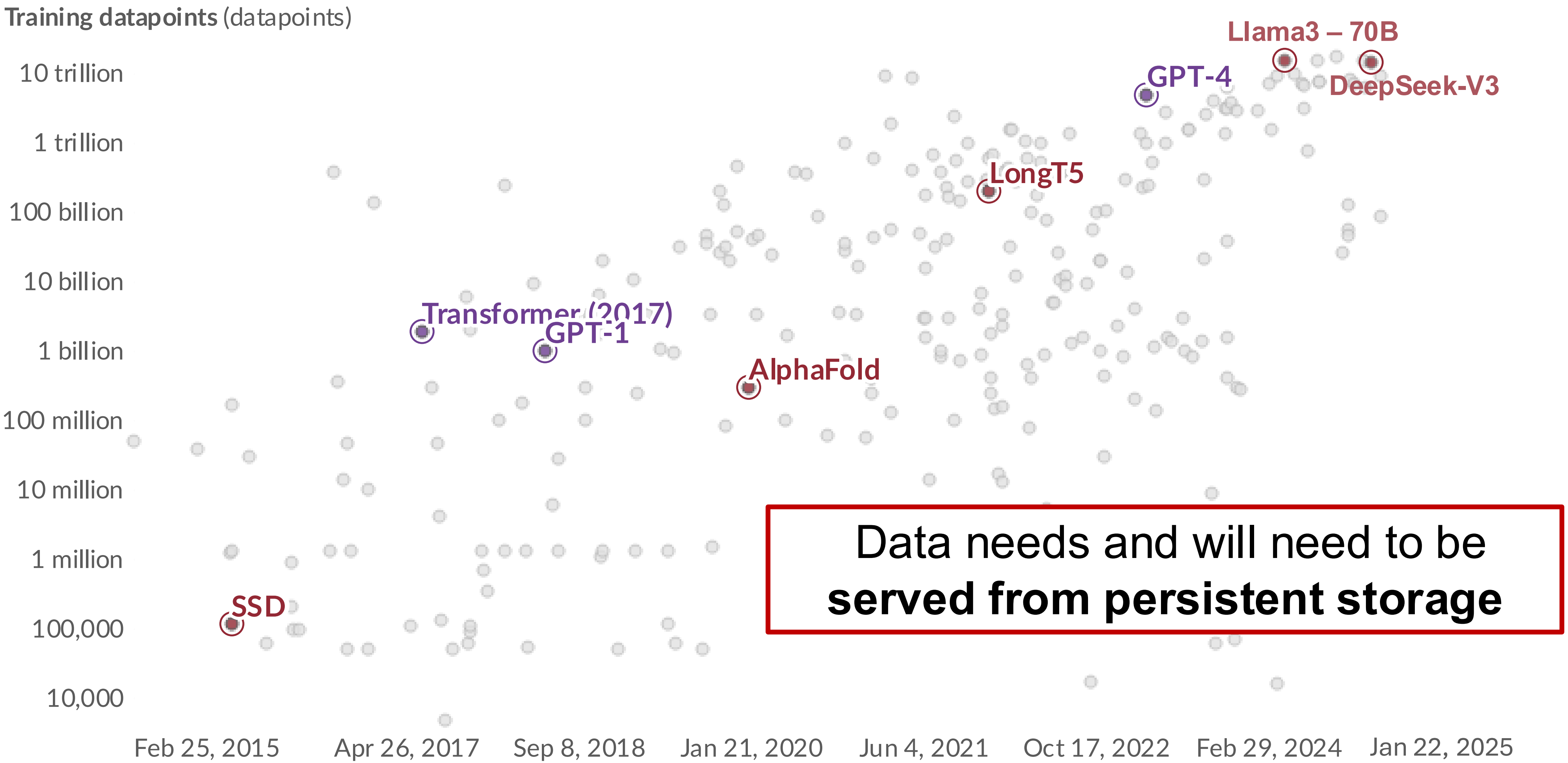
Model Sizes Growing Exponentially



If we include **training checkpoints, optimizer states, and multiple precision versions**, the full footprint of these models, including all stages of development, can easily reach **10-100+ TB** in storage.

Source: https://ourworldindata.org/grapher/exponential-growth-of-parameters-in-notable-ai-systems

Datasets Growing Exponentially



Data needs and will need to be **served from persistent storage**

Source: https://ourworldindata.org/grapher/exponential-growth-of-datapoints-used-to-train-notable-ai-systems

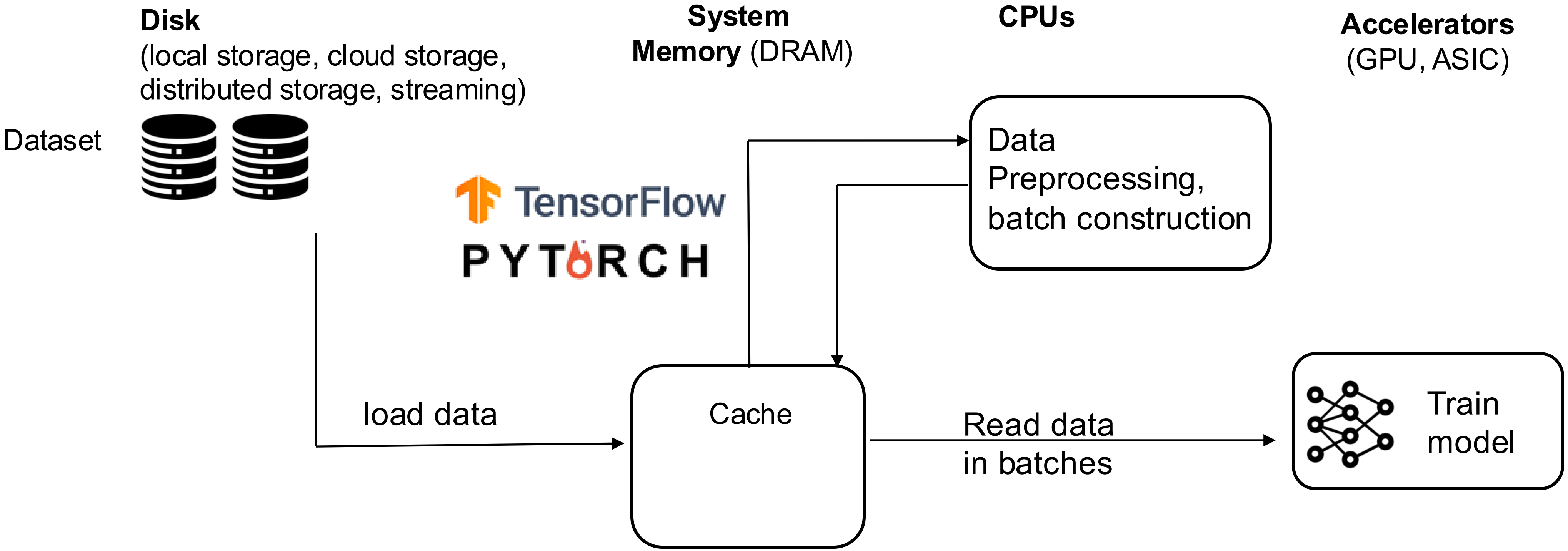
Data is the moving force of ML algorithms

… but in many projects the **storage decision is an afterthought**

Data pipeline in ML: Training

Storage resources

Compute resources



Disk
(local storage, cloud storage, distributed storage, streaming)

System Memory (DRAM)

CPUs

Accelerators
(GPU, ASIC)

Dataset

TensorFlow

PYTORCH

Data Preprocessing, batch construction

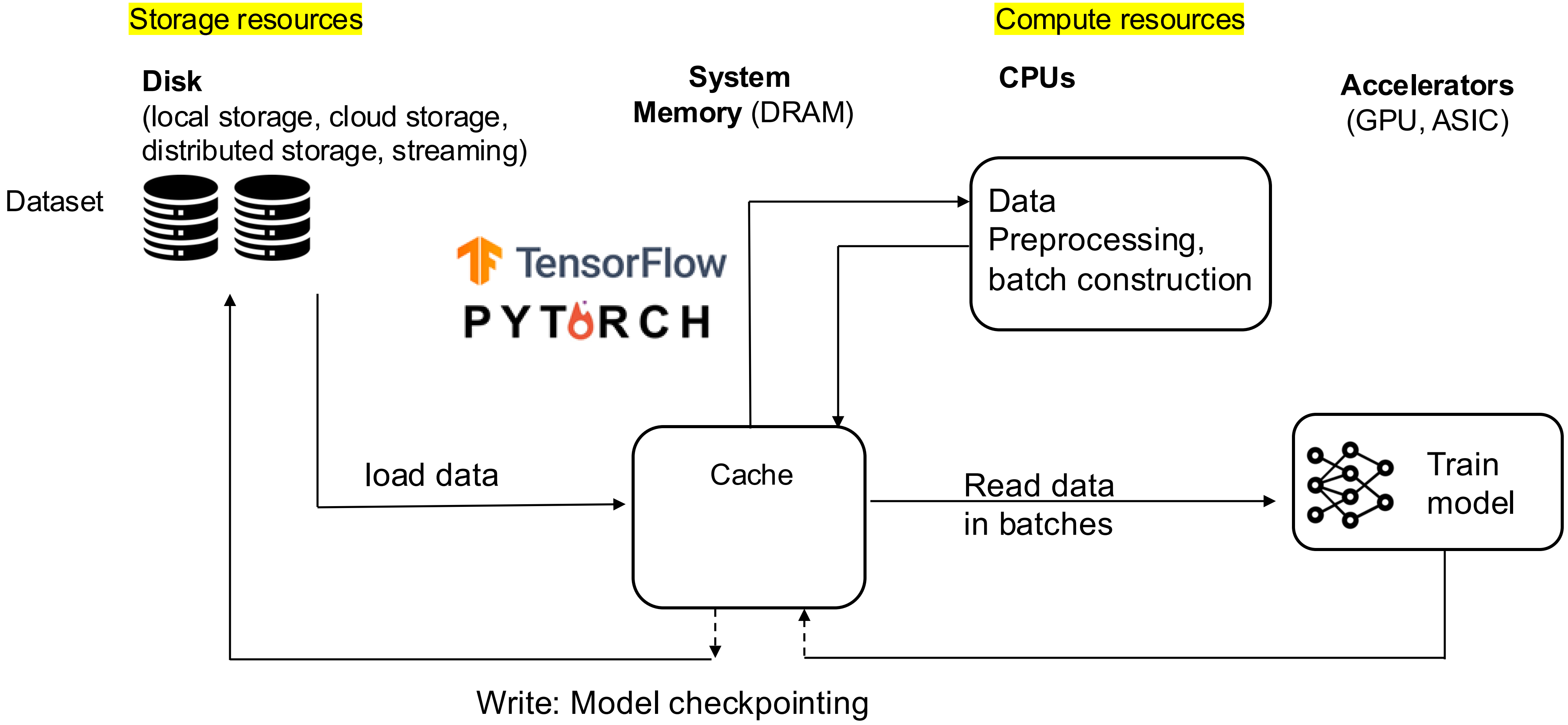
load data

Cache

Read data in batches

Train model

Data pipeline in ML: Training



Storage resources

Compute resources

Disk
(local storage, cloud storage, distributed storage, streaming)

System Memory (DRAM)

CPUs

Accelerators
(GPU, ASIC)

Dataset

TensorFlow

PYTORCH

Data Preprocessing, batch construction

load data

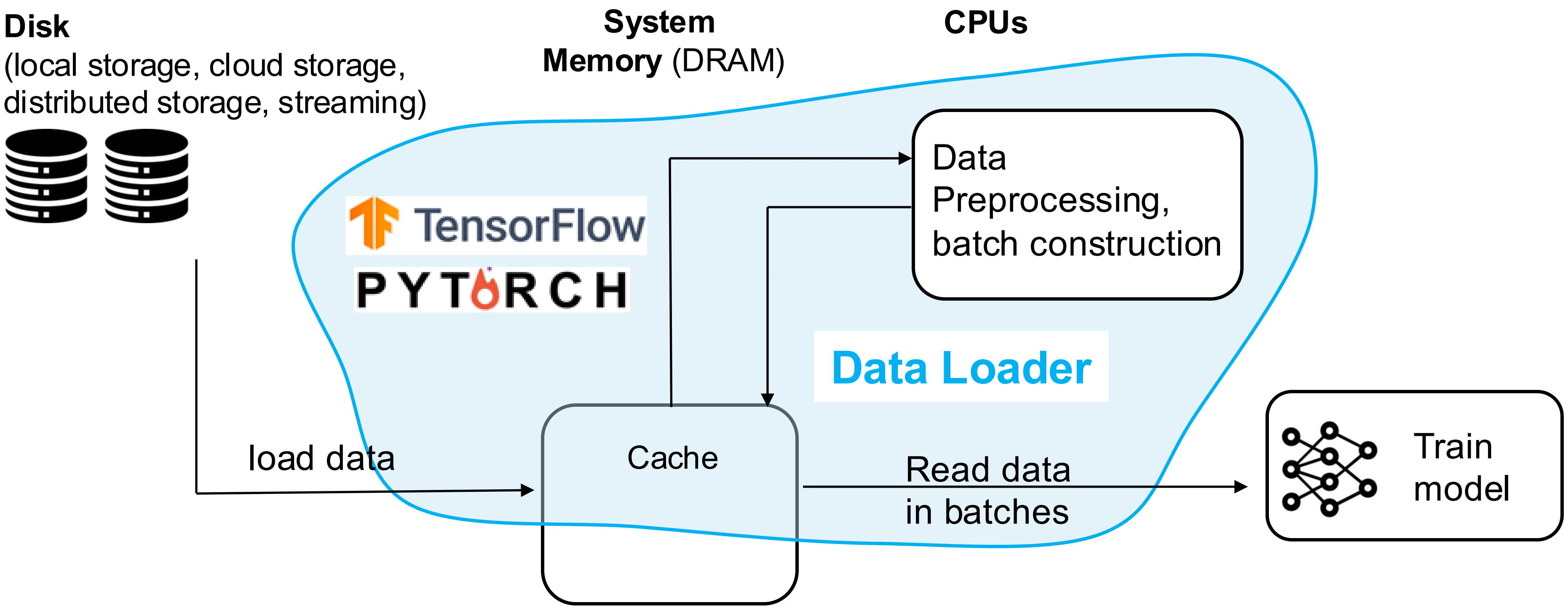
Cache

Read data in batches

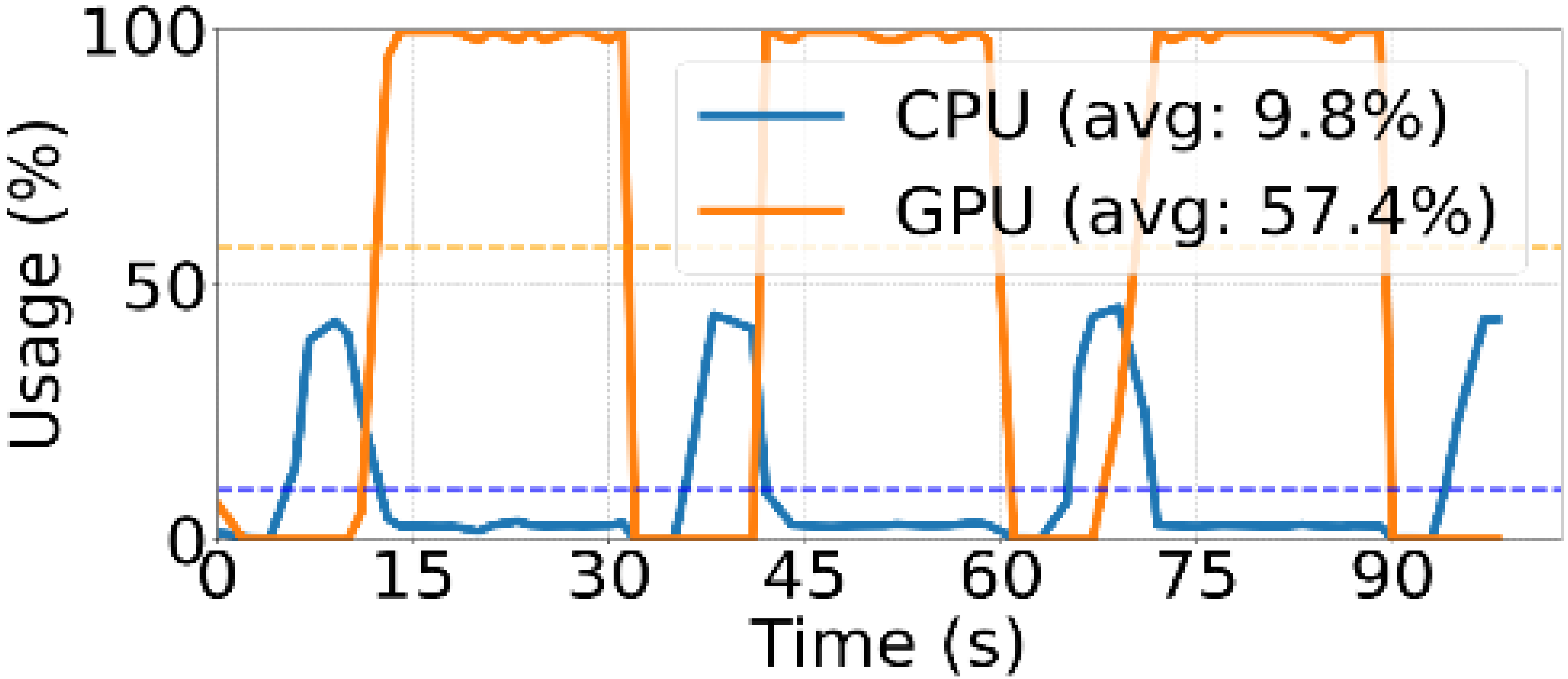
Train model

Write: Model checkpointing

Data preprocessing

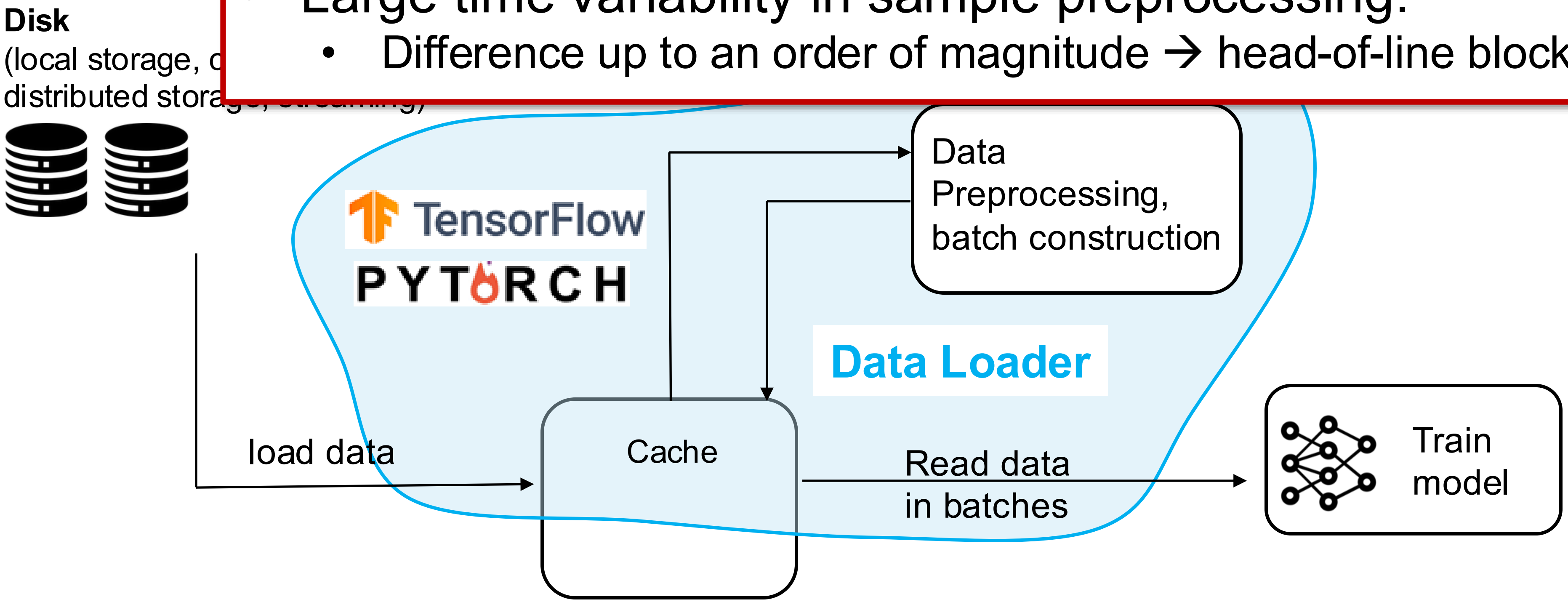


PyTorch Data Loader: Low GPU Use



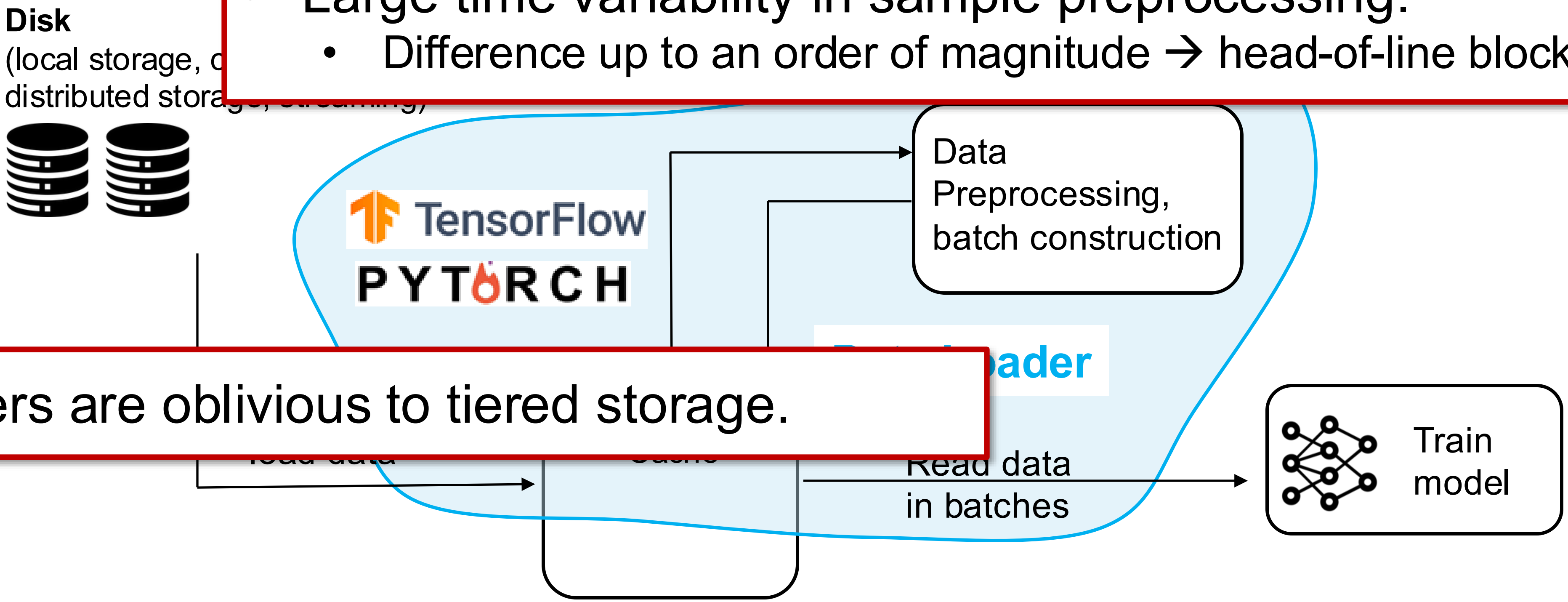
Data preprocessing

- Large time variability in sample preprocessing.
  - Difference up to an order of magnitude → head-of-line blocking



Data preprocessing

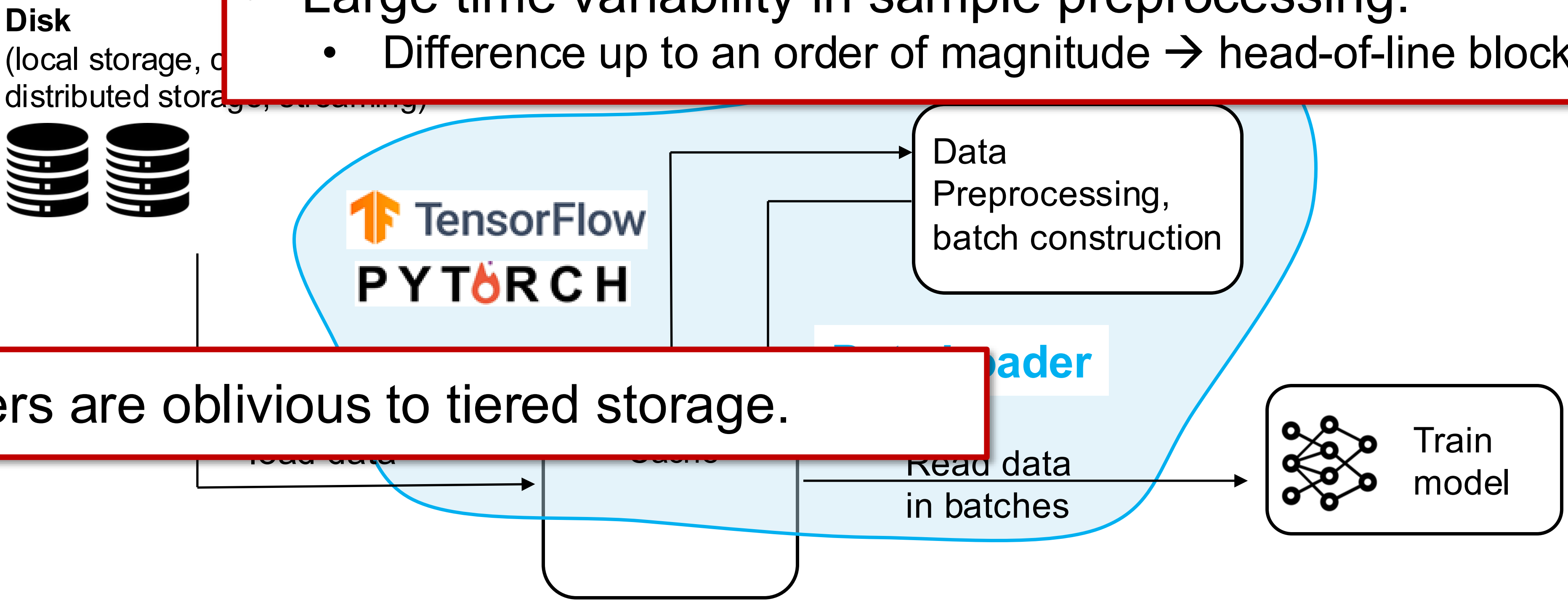
- Large time variability in sample preprocessing.
  - Difference up to an order of magnitude → head-of-line blocking



- Data loaders are oblivious to tiered storage.

Data preprocessing

- Large time variability in sample preprocessing.
  - Difference up to an order of magnitude → head-of-line blocking.

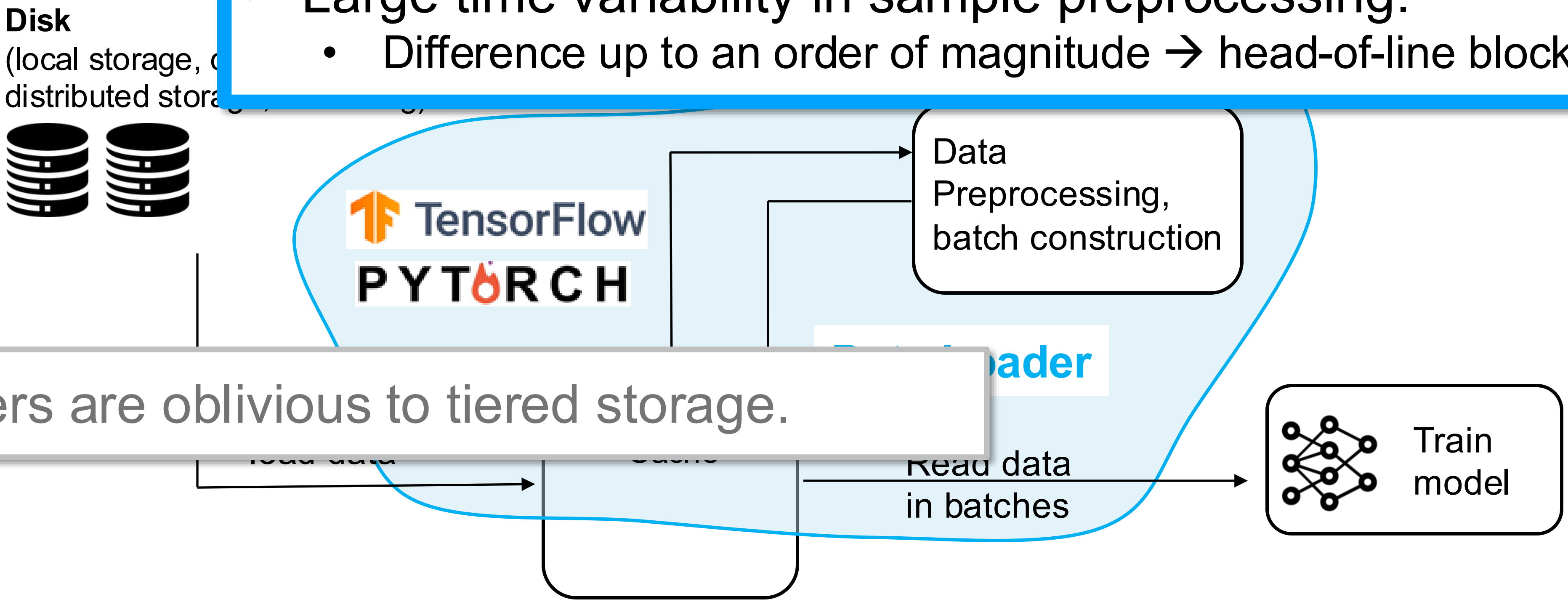


- Data loaders are oblivious to tiered storage.

- Dataset quality can slow down training.
  - Deduplication helps with training but is slow and memory intensive.

Data preprocessing

- Large time variability in sample preprocessing.
  - Difference up to an order of magnitude → head-of-line blocking.



- Data loaders are oblivious to tiered storage.

- Dataset quality can slow down training.
  - Deduplication helps with training but is slow and memory intensive.

SpeedyLoader: Efficient Pipelining of Data Preprocessing and Machine Learning Training

Rahma Nouaji[1], Stella Bitchebe[1], Ricardo Macedo[2], Oana Balmau[1]

1. McGill 2. INESC-TEC

Why do we care about data preprocessing?

- Data sample quality is crucial for prediction accuracy.
- Data preprocessing is often overlooked.
- Many workloads are randomly augmented for each batch
  - Need online data preprocessing.

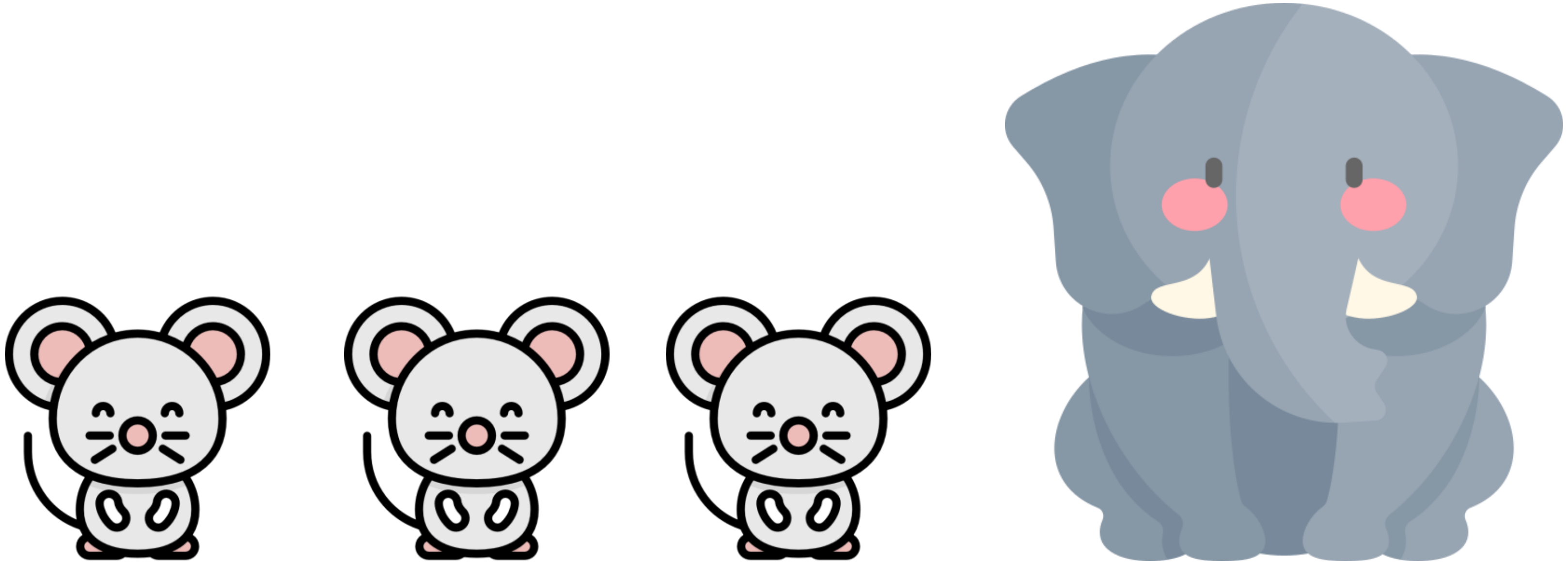
Why do we care about data preprocessing?

- Data sample quality is crucial for prediction accuracy.
- Data preprocessing is often overlooked.
- Many workloads are randomly augmented for each batch
  - Need online data preprocessing.

→ Used out-of-the-box, default data loaders do not efficiently pipeline CPU preprocessing with GPU training.

SpeedyLoader Key Insight

Large variability in the preprocessing time of different samples leads to head-of-line blocking.

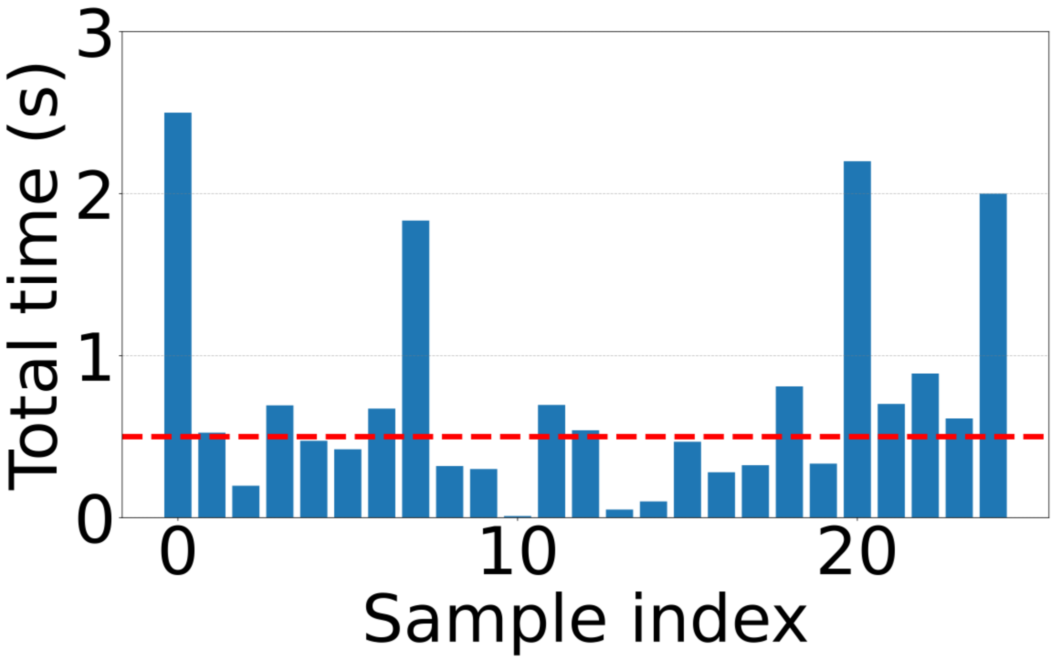


Head-of-line blocking caused by:

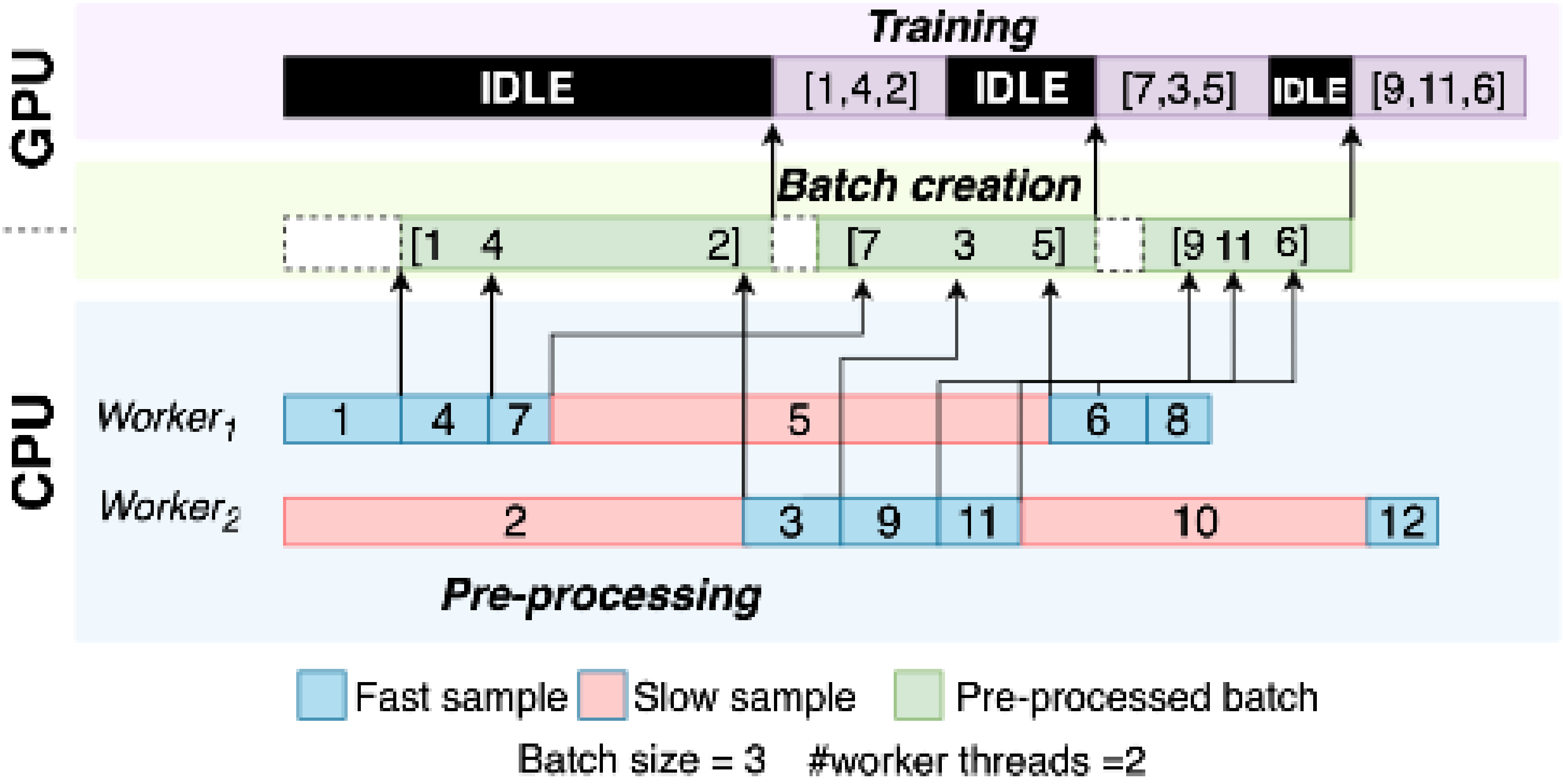
- Randomness in sample size,
- Information “richness”,
- Randomness inside preprocessing ops,
- Randomness in the op choice.

Example: 3D-Unet

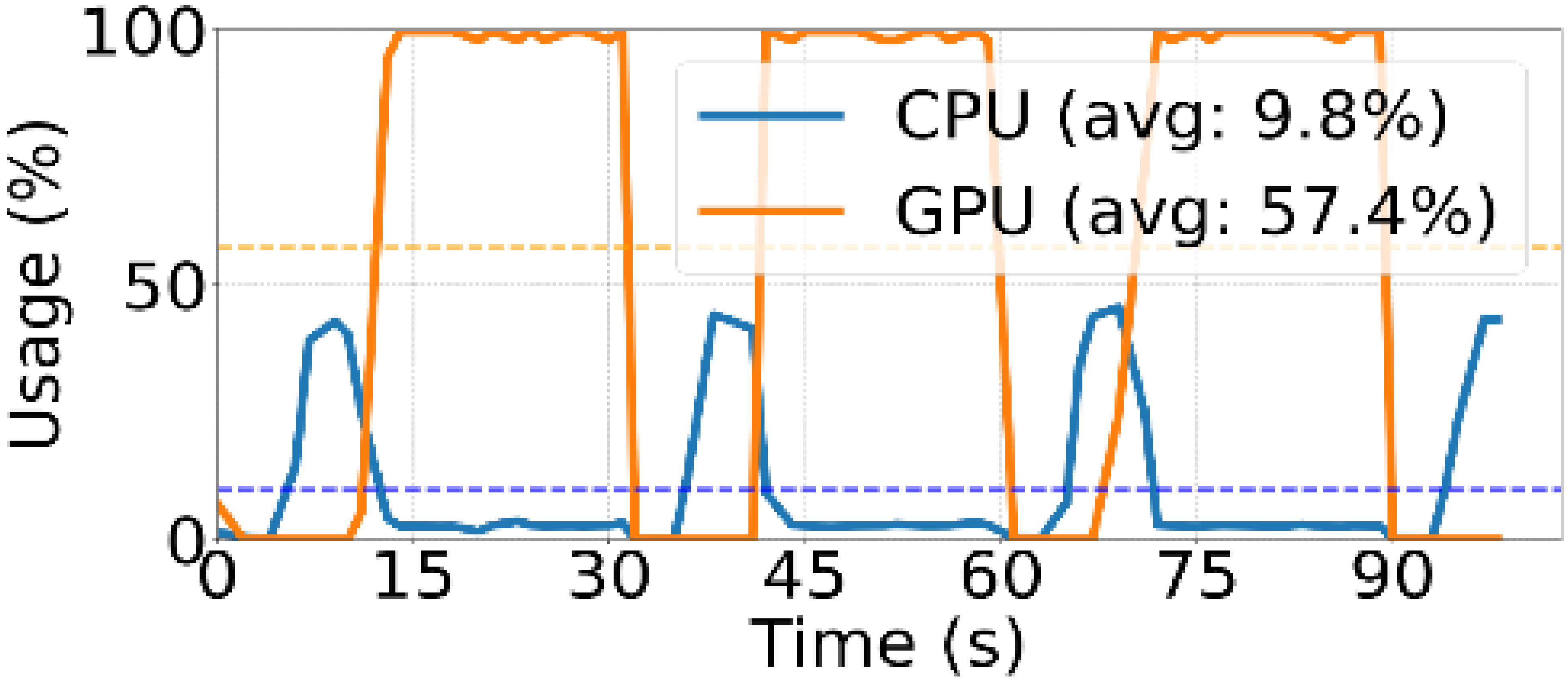
- The KiTS19 dataset
  - with 210 samples.
- 3D-UNet model.
- 8 data preprocessing steps.
- Dataset size: 29GB.



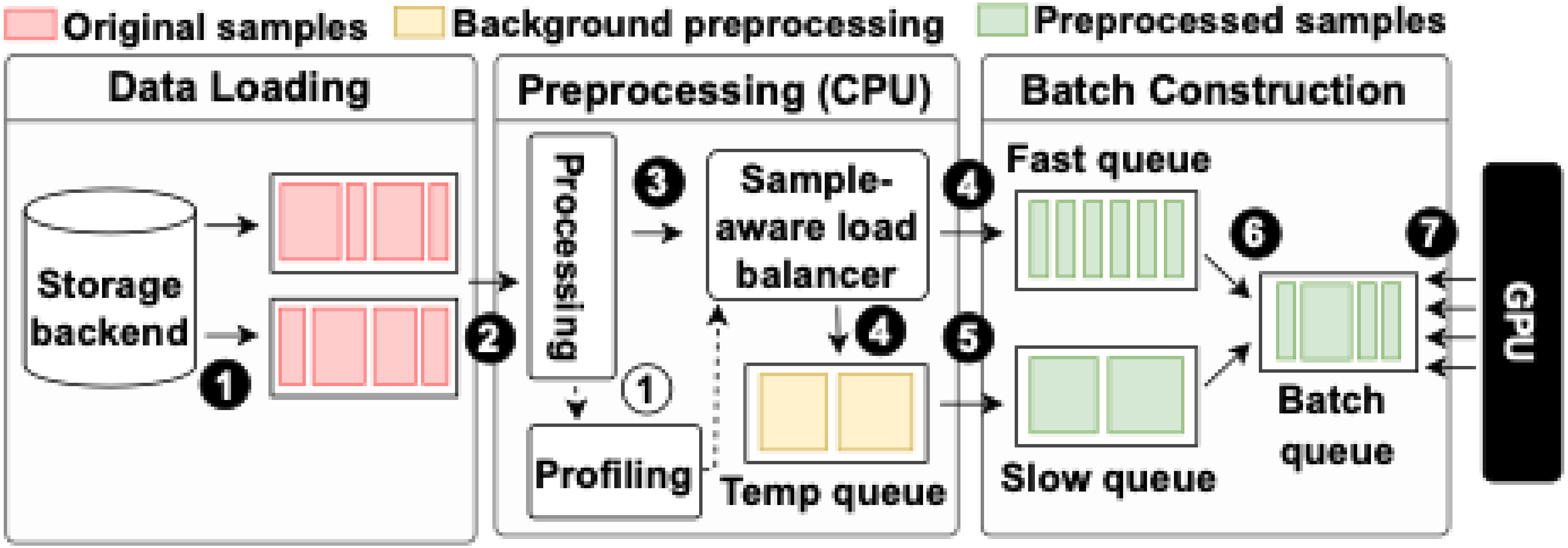
PyTorch Data Loader: Head-of-line Blocking



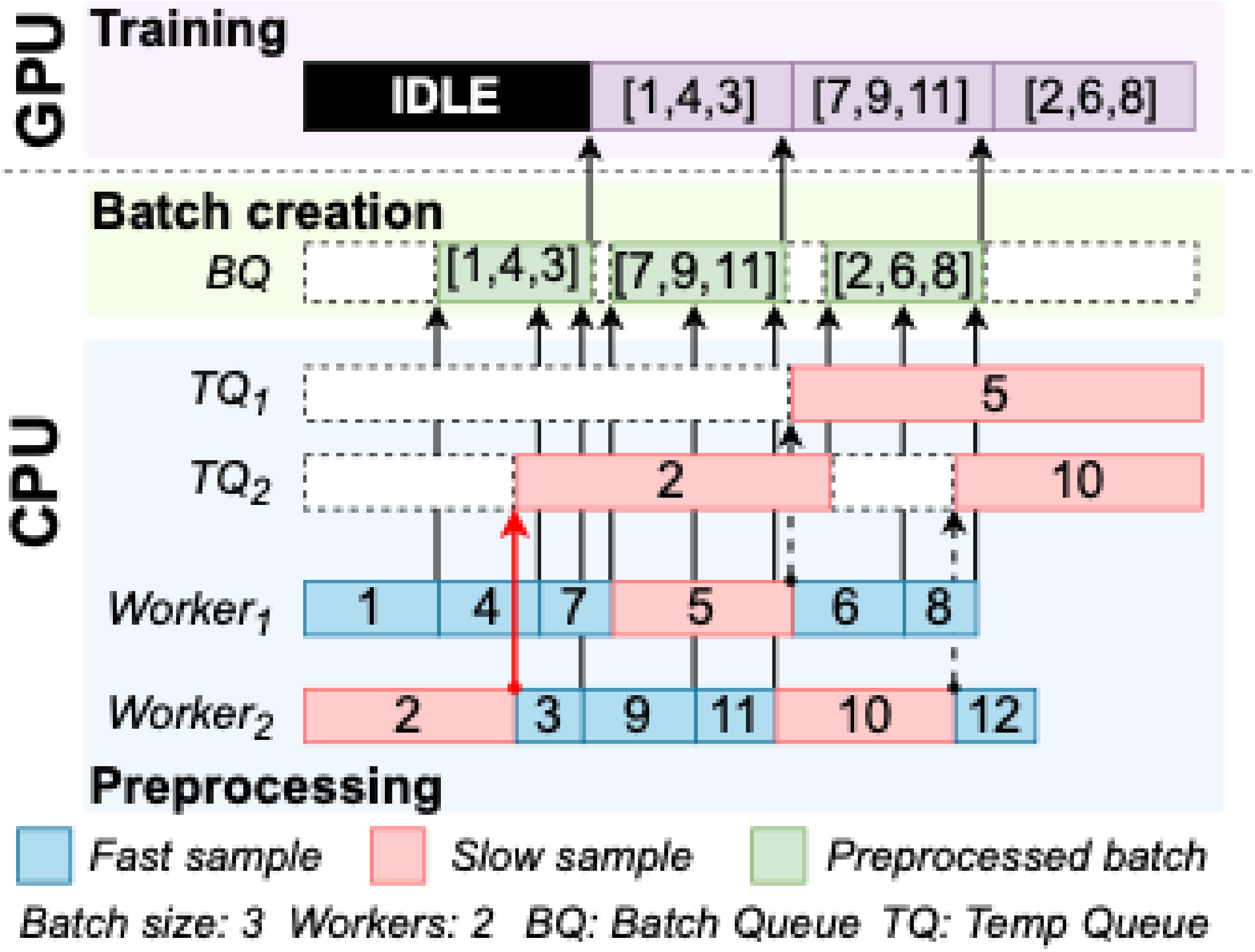
PyTorch Data Loader: Head-of-line Blocking



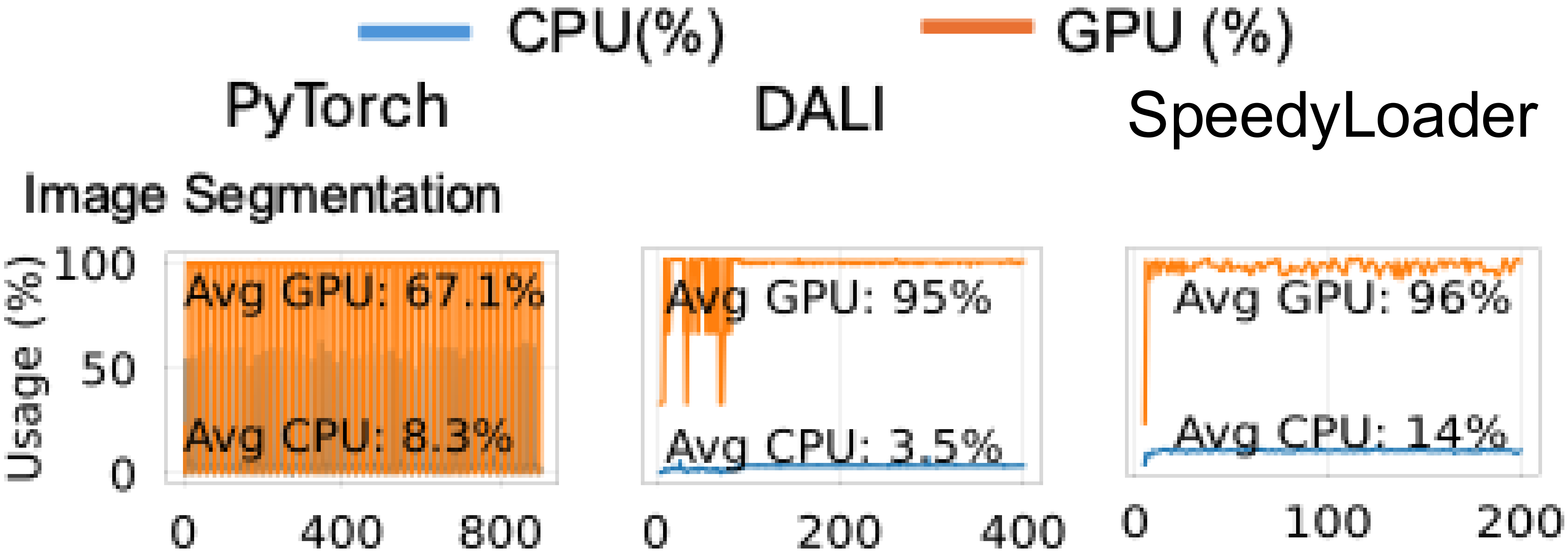
SpeedyLoader Design



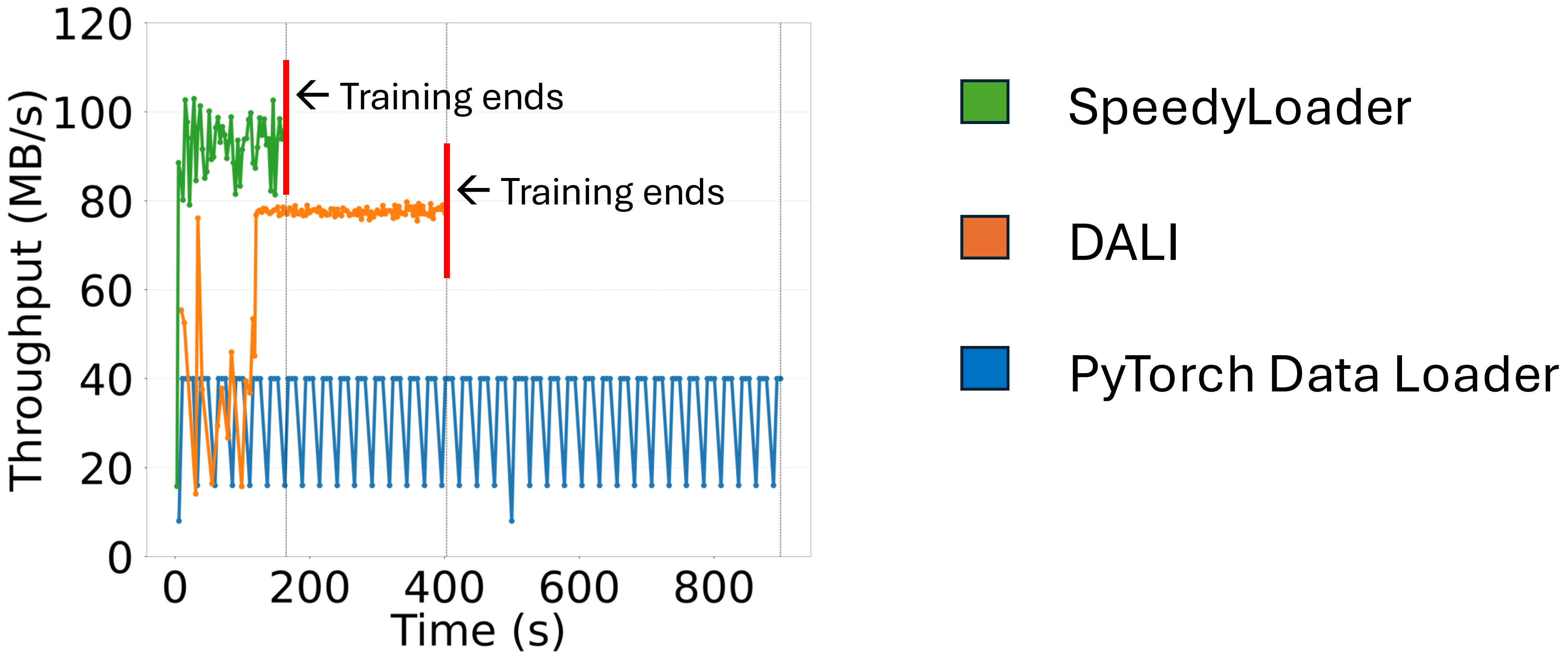
SpeedyLoader Mitigates Head-of-Line Blocking



GPU Use Improvement for 3D-Unet
4 x A100 GPU system



Throughput Improvement for 3D-Unet
4 x A100 GPU system



Key Takeaways

- Data preprocessing is important for efficient ML training.
- Preprocessing sample time can have order-of-magnitude variability.
- SpeedyLoader assesses fast and slow samples to avoid head-of-line blocking.

- *(Talk to me about **MLPerf Storage!**)*

ML Commons

Thanks to my postdoc and PhD students!

- Dr. Stella Bitchebe
- Rahma Nouaji
- Nelson Bore
- Jiaxuan Chen
- Shubham Vashisth
- Ruben Adao
- Pritish Mishra

https://discslab.cs.mcgill.ca

Thanks to MLPerf Storage co-chairs!

- Curtis Anderson, Hammerspace
- Dr. Huihuo Zheng, Argonne National Labs
- Johnu George, Nutanix

https://github.com/mlcommons/storage