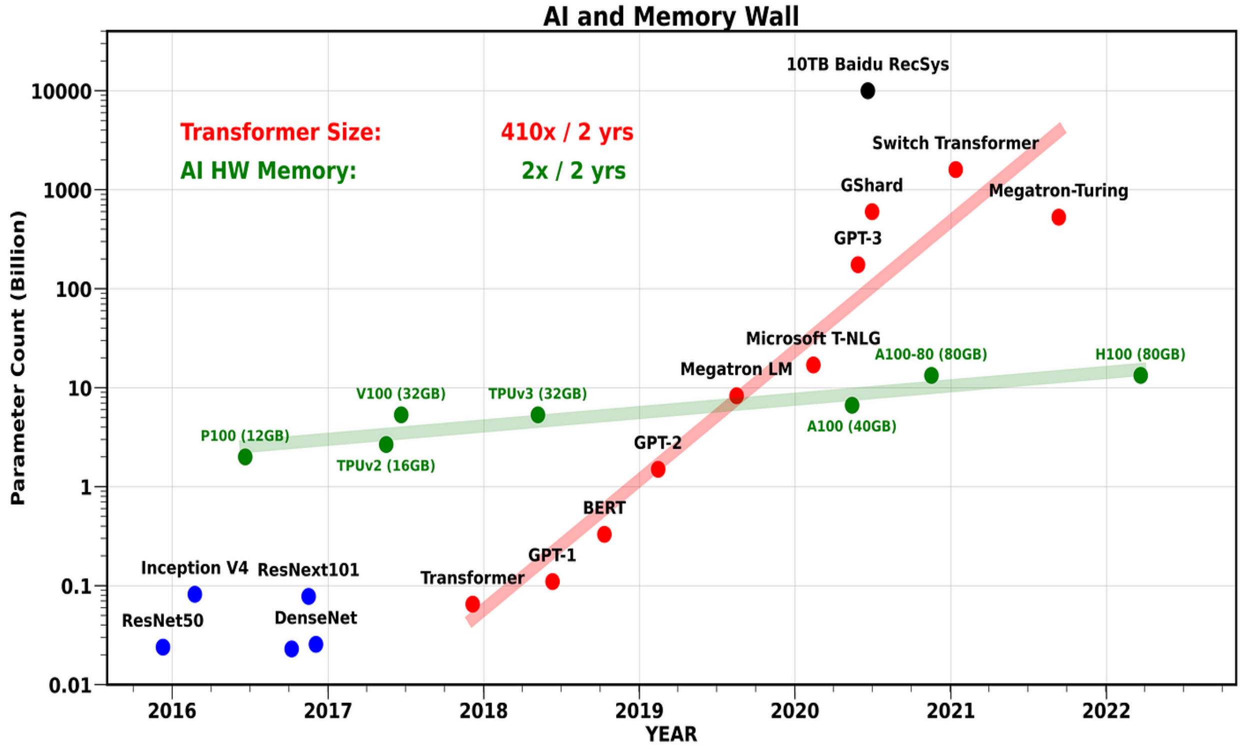
# Understanding Oversubscribed Memory Management for Deep Learning Training

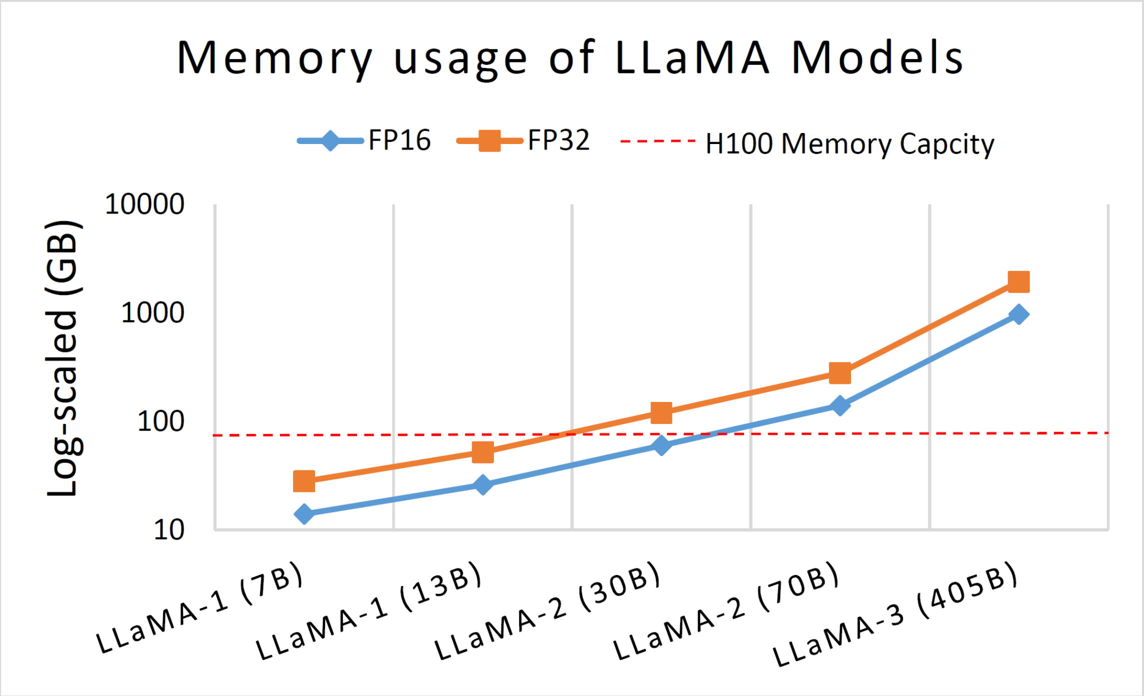
***Mao Lin*** *and Hyeran Jeon*

*University of California Merced*

March 31, 2025 · Rotterdam, Netherlands

# Backgrounds: GPU Memory Is No Longer Enough!




Model required memory: ↑ ↑ ↑ ↑ ↑
GPU memory capacity: ↑

Memory required(LLaMA-3(405B):
1.9TB = **24** * 80GB (H100)

Image source: https://arxiv.org/abs/2403.14123 https://www.substratus.ai/blog/llama-3-1-405b-gpu-requirements

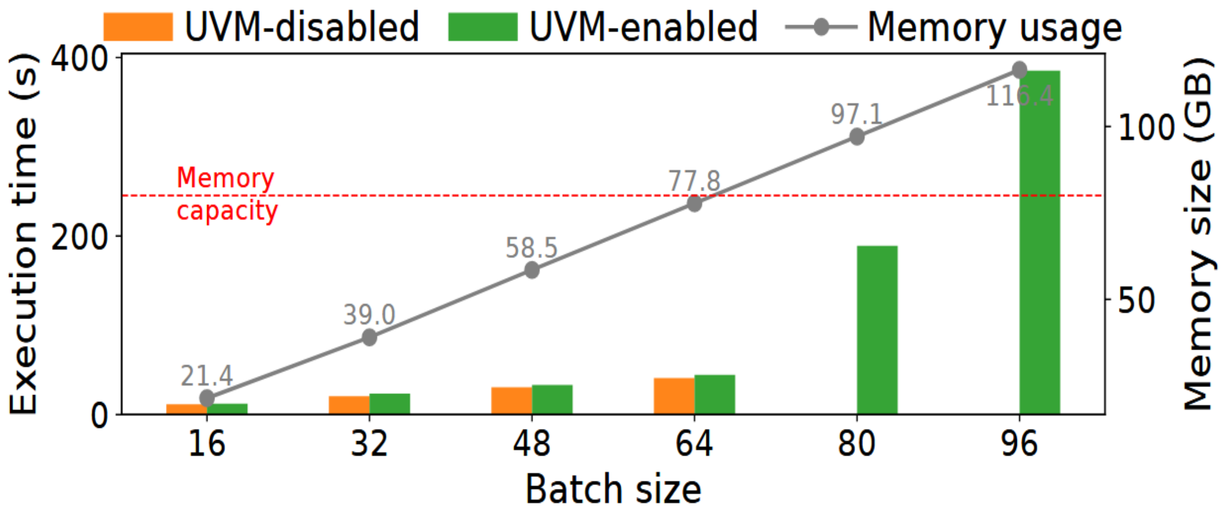
# Backgrounds: SOTA Solutions

- Multi-GPU Parallelism
  - GPU may not always be readily available
  - e.g., Megatron-LM@SC'21, MegaScale@NSDI'24, HAP@EuroSys'24, etc.
- Data Offloading/Checkpointing
  - Requires complex memory copy orchestration
  - e.g., ZeRO-Offload@ATC'21, ZeRO-infinity@SC'21, POET@PMLR'22, etc.
- Intermediate Result Recomputation
  - Introduces extra computation overhead
  - e.g., Skipper@MICRO'22, Aceso@EuroSys'24, AdaPipe@ASPLOS'24, etc.
- Memory compression & Quantization
  - Adds overhead and potential accuracy loss
  - e.g., ZeroQuant(4+2)@arXiv'24 (Deepspeed), FP6-LLM@arXiv'24 (Deepspeed), etc.

# UVM Can Be Another Solution!

- Unified Virtual Memory
  - On-demand migration
  - Memory oversubscription
    - Workloads' memory footprint > GPU memory capacity

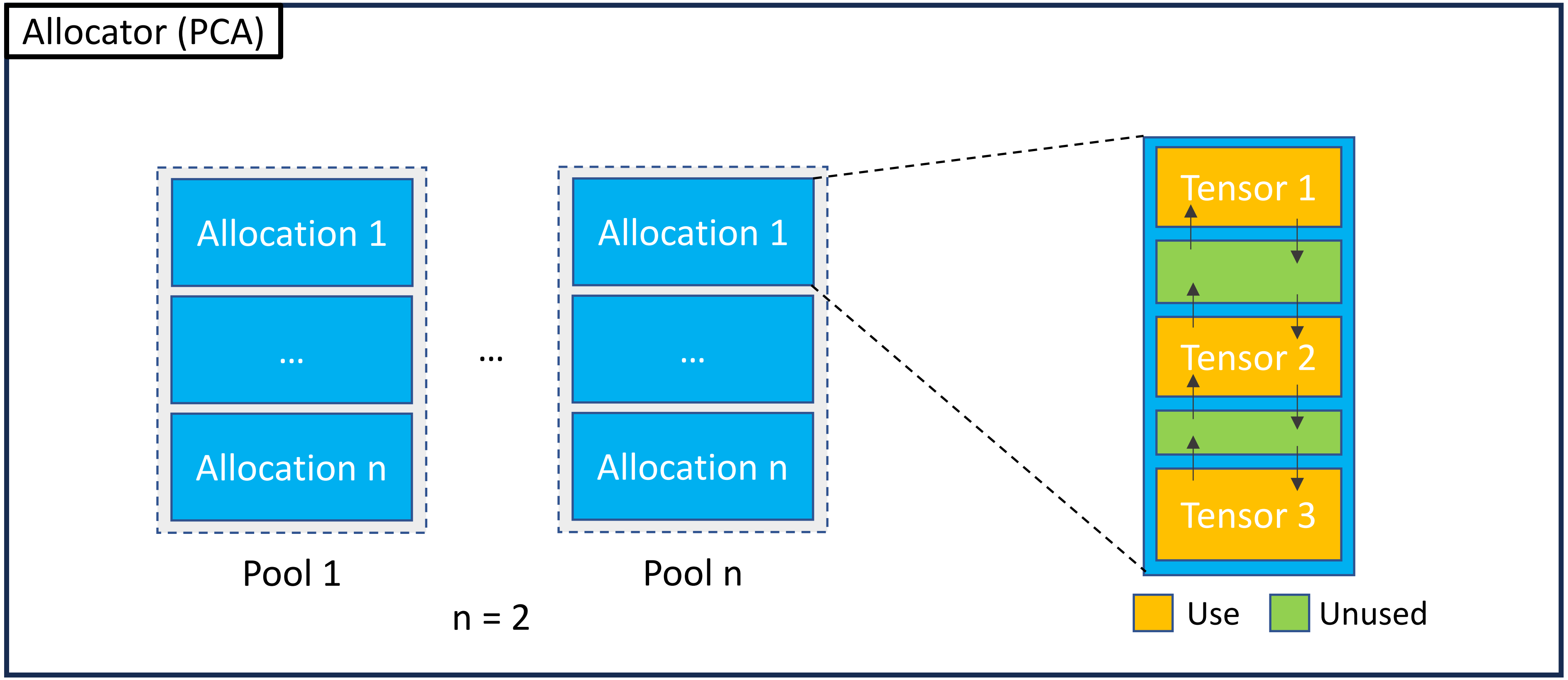
- The challenges
  - UVM overhead (page fault)
  - Interaction with DL framework




**GPT-2 performance w/ and w/o UVM and memory usage across varying batch sizes**

# PyTorch Caching Allocator (PCA)

- Memory hierarchy:
  - Allocator -> Pools -> Allocations -> Subranges (tensors)



# Goal of This Study

- *Explore the potential of using UVM for deep learning (DL) workloads*

- *Examine, for the first time, how DL frameworks' unique memory management (PCA) interacts with UVM*

- *Provide insights for efficiently adopting UVM in DL systems*

# Evaluation Platform & Targeting DL Models

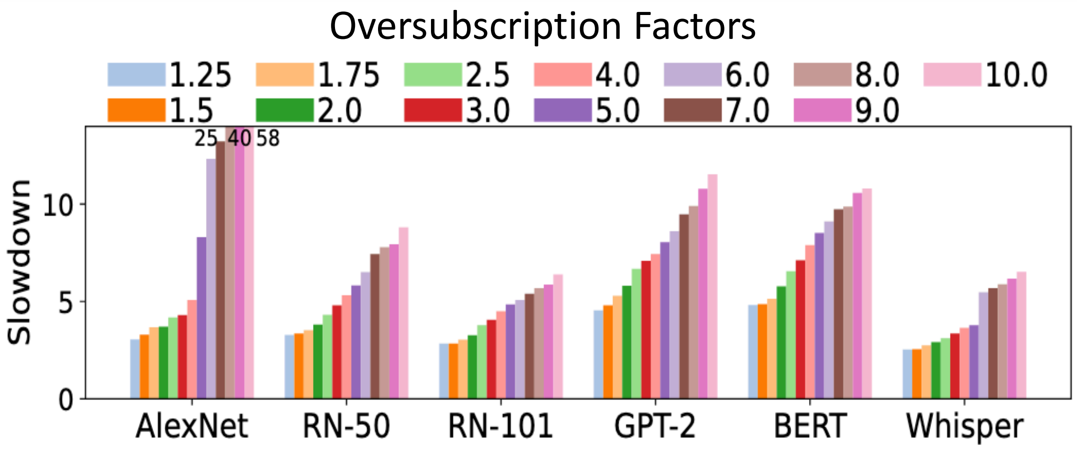
**Evaluation Platform**

| CPU | GPU | System | System Memory | GPU Driver | CUDA Toolkit | Nsight Systems |
|---|---|---|---|---|---|---|
| Intel(R) Xeon(R) Gold 5320 | NVIDIA A100 80GB PCIe | Linux 5.14 | 128 GB | 550.90.12 | 12.1 | v.2023.1.2 |

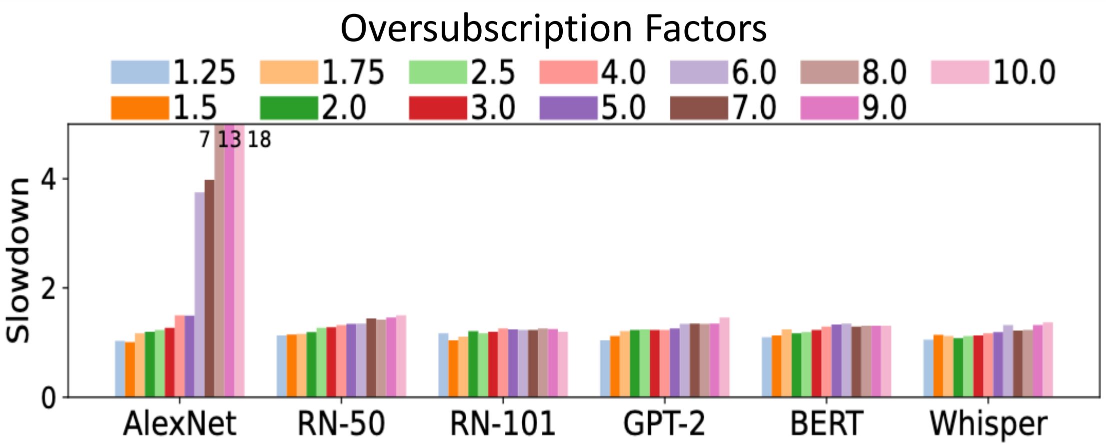
**Evaluated DL Models**

| Model | Type | Layers | Architecture | Batch Size | Memory Footprint (MB) |
|---|---|---|---|---|---|
| AlexNet | CNN | 8 | Convolutional Full Connected | 128 | 5316 |
| ResNet50 | CNN | 50 | Residual Block | 32 | 15952 |
| ResNet101 | CNN | 101 | Residual Block | 32 | 22588 |
| GPT-2 | Transformer | 12 | Transformer (Decoder) | 8 | 12008 |
| BERT | Transformer | 12 | Transformer (Encoder) | 16 | 12350 |
| Whisper (small) | Transformer | 12 | Transformer (En/De-coder) | 16 | 9824 |

# UVM Is Good for DL Workloads

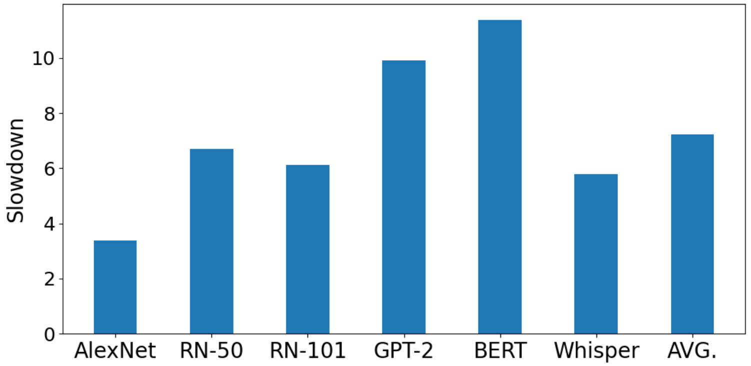



**(a)** With PCA.



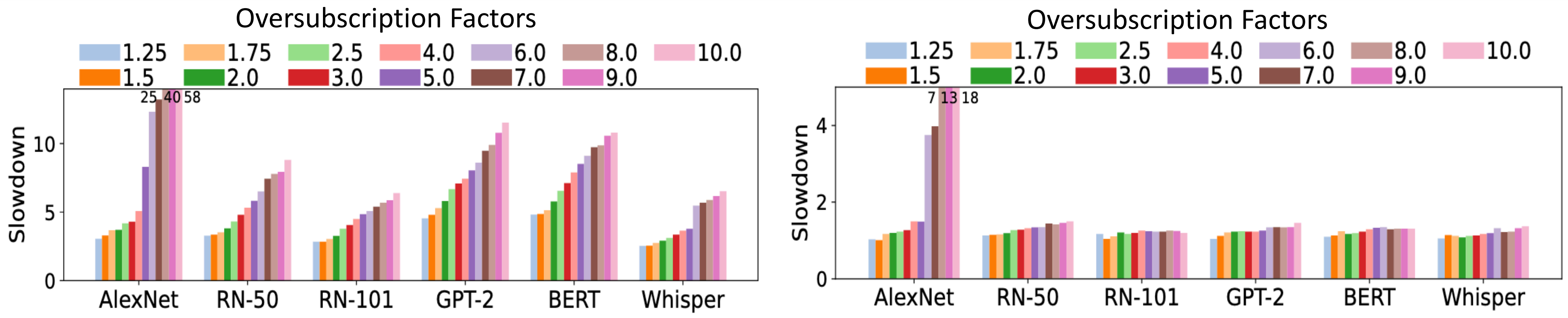

**(b)** Without PCA.

**Performance under diverse oversubscription factors.**




**Non PCA baseline vs. PCA baseline.**

# UVM Is Good for DL Workloads



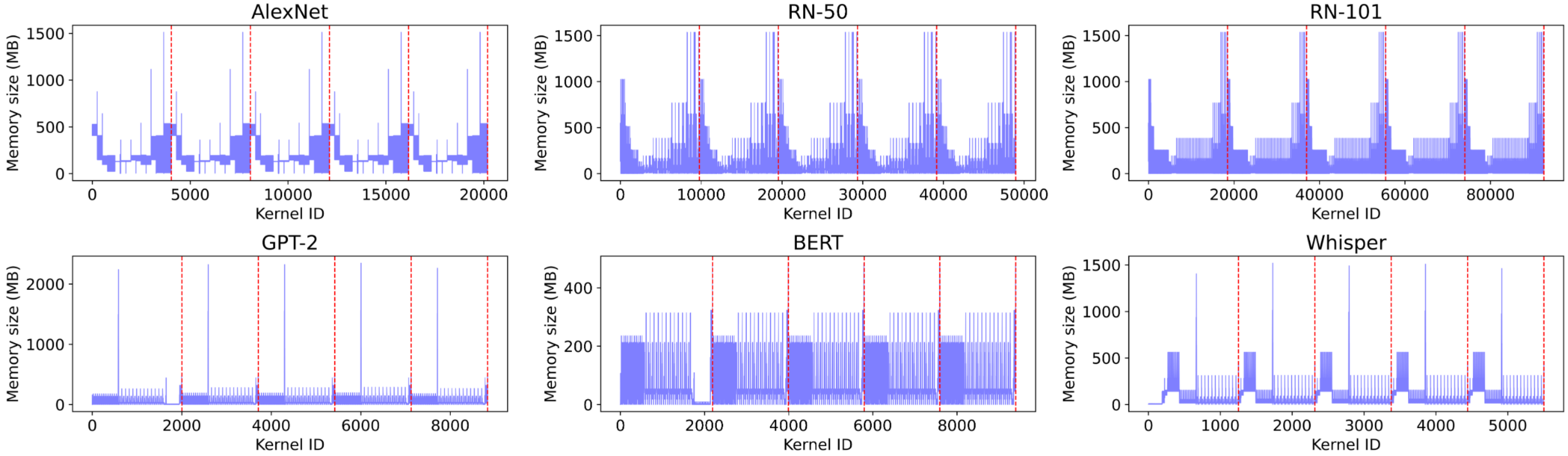

**(a)** With PCA. **(b)** Without PCA.

**Performance under diverse oversubscription factors.**

***<u>Observation 1</u>****: LLMs, such as GPT-2 and BERT, benefit more from UVM than simpler CNNs under a high oversubscription factor, due to intensive computation overlapping with page fault handling.*

***<u>Observation 2</u>****: Despite recommendations to limit the oversubscription factor to 1.25, our findings show acceptable overhead even at higher values for DL workloads.*

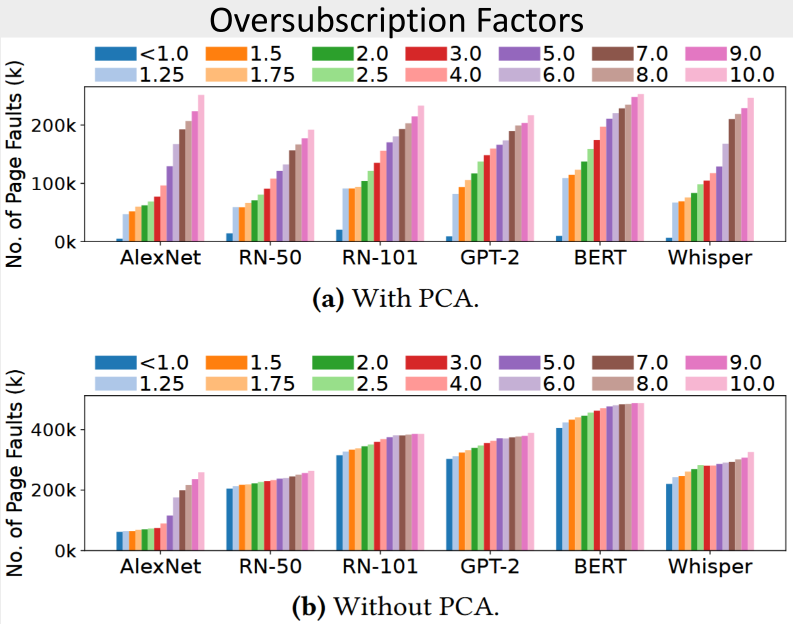
# Why?



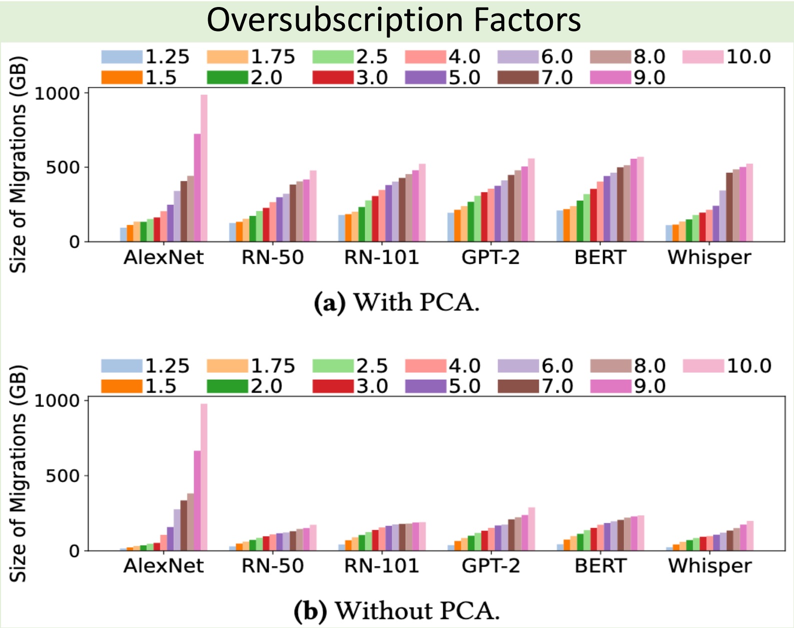

**Memory usage per kernel over time for various models.**

The memory footprint is large, but memory usage per kernel is not that large.

# PCA Trades Pages Faults for Migrations




Number of pages faults.




Size of Migrations.

***<u>Observation 3</u>**: PCA's pool-based memory management effectively reduces substantial page faults. Given that the main performance bottleneck of UVM is expensive page faults, pool-based memory management can be a solution.*

***<u>Observation 4</u>**: PCA trades page fault overhead for memory migration overhead. As UVM's smart prefetching and pre-eviction mechanisms effectively remove memory migrations from the critical path, the cumbersome page fault overhead of UVM can be tackled by integrating the UVM with PCA.*

# Takeaways

- *DL workloads' memory behaviors suit UVM, enabling large-scale execution on limited GPU memory without needing multiple GPUs.*

- *UVM, once seen as inefficient for DL due to page faults, benefits from modern techniques like PCA.*

- *UVM with PCA is effective, but further DL-specific, context-aware optimizations (prefetching/pre-eviction) can enhance performance.*

UCMERCED

# Thank you!

# Any questions?