***Papers Reviewed***

# Pre Training-

# GEMINI: Fast Failure Recovery in Distributed Training with In-Memory Checkpoints-To address these two challenges, this paper proposes: 1) *a provably near-optimal checkpoint placement strategy* to maximize the probability of failure recovery from checkpoints in CPU memory; and 2) *a checkpoint traffic scheduling algorithm* to minimize, if not eliminate, the interference of checkpoint traffic on model training. Our evaluation shows that overall Gemini achieves a faster failure recovery by more than 13× than existing solutions.

# The Llama 3 Herd of Models- We find that Llama 3 delivers comparable quality to leading language models such as GPT-4 on a plethora of tasks. Our largest model is a dense Transformer with 405B parameters and a context window of up to 128K tokens

# Serving-

# Orca-Iteration-level scheduling, a new scheduling mechanism that schedules execution at the granularity of iteration (instead of request) where the scheduler invokes the execution engine to run only a single iteration of the model on the batch. Single iteration of the model on the batch. In addition, to apply batching and iteration-level scheduling to a Transformer model at the same time, we suggest selective batching, which applies batching only to a selected set of operations

# LLM in flash- "Windowing" strategically reduces data transfer by reusing previously activated neurons, and second, "row-column bundling", tailored to the sequential data access strengths of flash memory, increases the size of data chunks read from flash memory.

# BlockLLM- BlockLLM, a serving system that exploits the potential of sharing components among fine-tuned LLM models to offer an efficient and flexible solution for LLM workloads. BlockLLM consists of an offline block zoo, for storing the blocks, and an online system to serve the requests through chains of blocks.

# Multi-Model Systems-

# DISTMM: Accelerating distributed multimodal model training- DISTMM exploits the heterogeneity among submodules, applying different distributed parallelism strategies for each submodule, e.g., using Tensor Parallelism for a computation-intensive submodule, and Data Parallelism for a submodule with a small number of parameters.

# Optimus: Optimus, a distributed MLLM training system that reduces end-to-end MLLM training time. Optimus is based on our principled analysis that scheduling the encoder computation within the LLM bubbles can reduce bubbles in MLLM training. To make scheduling encoder computation possible for all GPUs, Optimus searches the separate parallel plans for encoder and LLM, and adopts a bubble scheduling algorithm to enable exploiting LLM bubbles without breaking the original data dependencies in the MLLM model architecture.

# LLM for Systems-

# Large Language Models for Compiler Optimization- We present a 7B-parameter transformer model trained from scratch to optimize LLVM assembly for code size. The model takes as input unoptimized assembly and outputs a list of compiler options to best optimize the program.

# The Hitchhiker's Guide to Program Analysis: A Journey with Large Language Models- , LLift demonstrates a potent capability, showcasing a reasonable precision (50%) and appearing to have no missing bugs. It even identified 13 previously unknown UBI bugs in the Linux kernel. This research paves the way for new opportunities and methodologies in using LLMs for bug discovery in extensive, real-world datasets. , LLift demonstrates a potent capability, showcasing a reasonable precision (50%) and appearing to have no missing bugs. It even identified 13 previously unknown UBI bugs in the Linux kernel. This research paves the way for new opportunities and methodologies in using LLMs for bug discovery in extensive, real-world datasets.