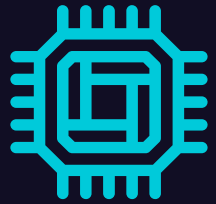
Optimizing AI Workloads in Linux-based Heterogeneous Systems



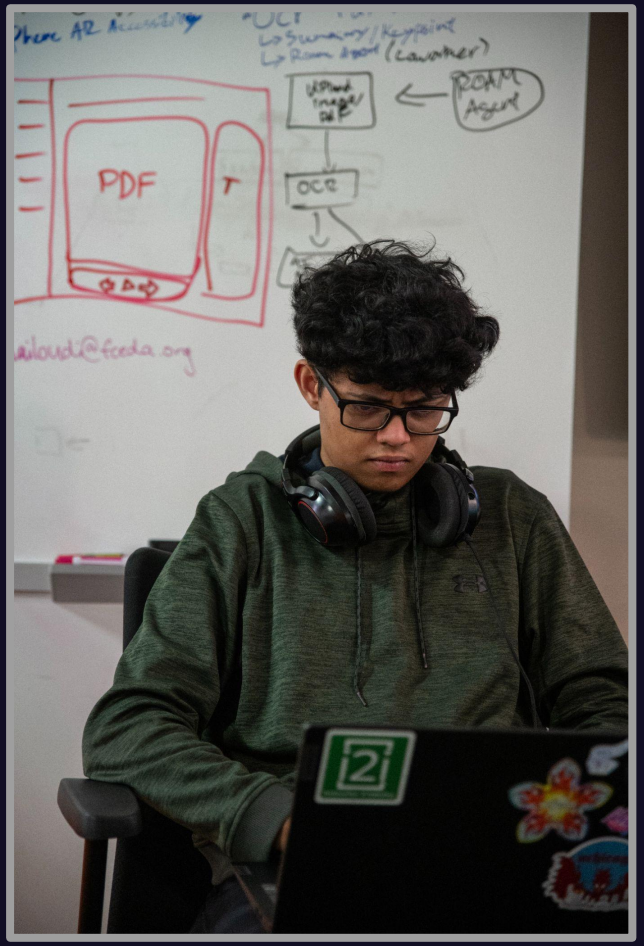
Sami F. Elsayed

The ENEE331 Computer Systems Lab - 2025

21th of May 2025

About Me

- Senior, Lead Sysadmin at tjCSL
  - Worked on Ion, Cluster, Workstations, & Networking
- Future Computer Engineering major at George Mason University
- Inspired by the challenge of optimizing AI performance on diverse hardware

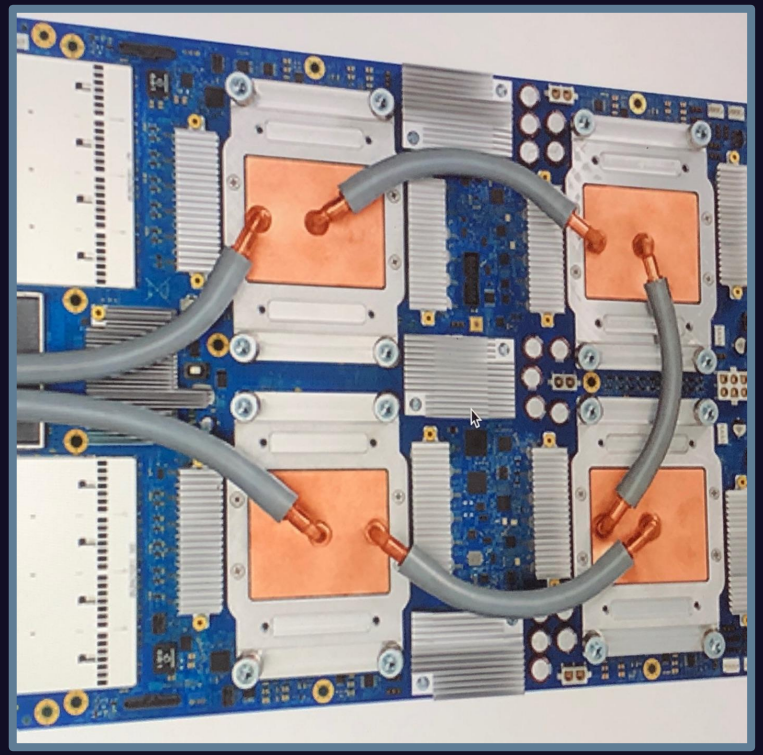


Project Motivation

- AI workloads demand high performance, energy efficiency, and thermal stability.
- Heterogeneous systems (CPUs, GPUs) offer potential but face inefficiencies.
- Goal: Develop a unified framework to optimize AI workloads in Linux.

Background Info

- **Processors**
  - *CPU*: General-purpose, sequential processing.
  - *GPU*: Parallel processing, ideal for AI tasks.
  - *TPU*: Made by Google, specific-purpose, ideal for ML/deep learning tasks.
- **Heterogeneous Systems**
  - Combination of different processors types (CPU + GPU).
- **AI Workloads**
  - Training models like neural networks, image classification, etc.
- **Linux**
  - The chosen OS for this project.
  - Ideal for custom performance tuning.

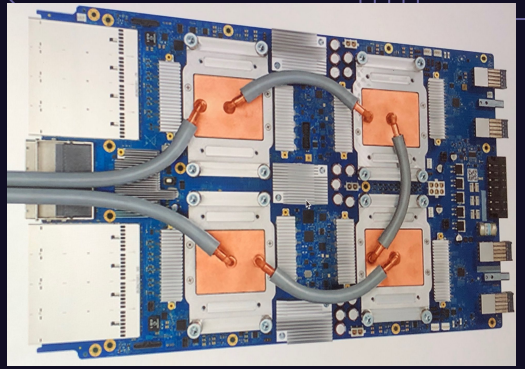


Central Processing Unit (CPU)

- General-purpose processor for a wide range of tasks.
- Few powerful cores; optimized for sequential processing.
- Great for tasks requiring single-threaded performance.

Processing Unit (GPU)

- Optimized for parallel processing, originally for graphics.
- Thousands of cores, ideal for large datasets and matrix operations.
- Best for tasks requiring high parallelism (e.g., deep learning).



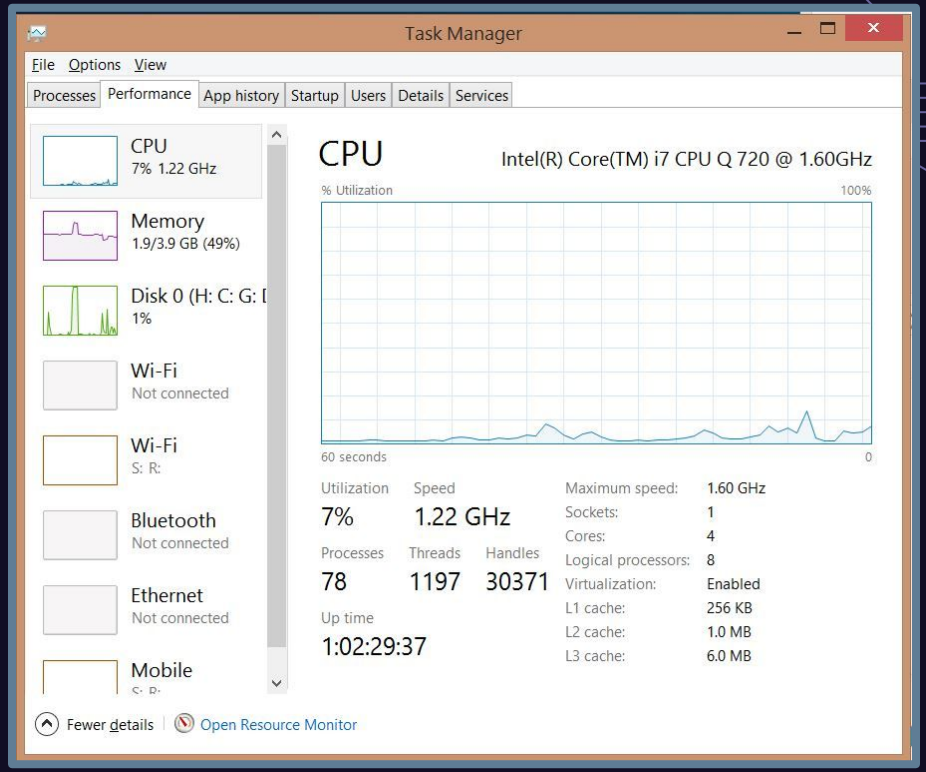
Tensor Processing Unit (TPU)

- Specialized for deep learning and tensor operations.
- Optimized for high-efficiency matrix calculations.
- Extremely fast for neural network tasks, but limited versatility.

Processors

Problems

- **Slow Running Time** of AI models on conventional hardware.
- **High Power Consumption** during model training and inference.
- **Thermal Management issues**, particularly in high-performance settings.



Phase 1

Data Collection & Baseline Testing

Phase 1: Initial Data Collection & Baseline Testing

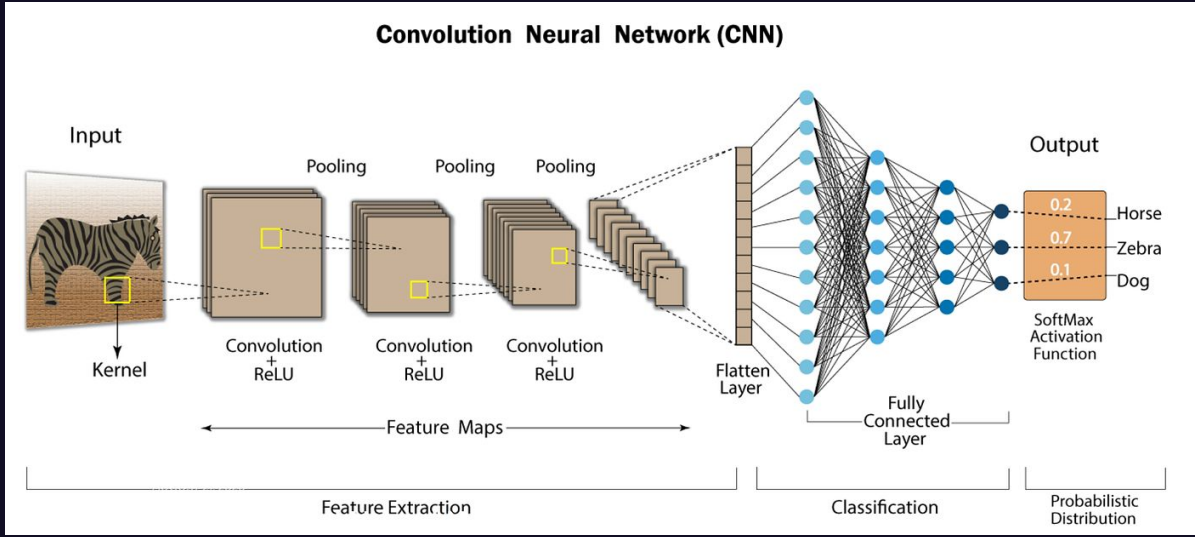
- **Goal**: Establish baseline performance for AI workloads on heterogeneous systems.
- **To do**:
  - Run workloads (matrix multiplication, CNNs) using common frameworks (TensorFlow). Measuring energy consumption, running time, thermal performance.
- **Tools**: Use profiling tools like `nvidia-smi` (for GPU), `perf`, `htop`, and energy consumption monitors.

Setup

- Workstation “Duke”
- AMD Ryzen 7 17000
- dual NVIDIA 1080 Ti GPUs
- Ubuntu 22.04

Convolutional Neural Network (CNN)

- Type of Deep Learning model
- Highly effective for processing and analyzing data with a grid-like structure
  - Popularly used for images and videos!



Phase 2

Dual Optimization Path

Path 1: Software (Workload) Optimization

Phase 2: Dual Optimization Path

Once baseline data is collected, the project will moves into two optimization paths:

Path 1: Software Optimization

- **Goal**: Optimization of the AI code to achieve better performance. Reduce running time, energy consumption, making AI code more efficient, not changing the hardware.
- **To do:**
  - Apply **algorithmic improvements** to AI models (i.e layer-wise computation in CNNs).
  - Explore AI model **compression techniques** (i.e pruning, quantization).

Phase 2: Dual Optimization Path (cont.)

Path 2: System and Hardware Optimization

- **Goal**: Optimize system configurations & hardware to enhance performance.
- **To do**:
  - Tuning CPU/GPU configuration.
  - Tweak Linux kernel parameters, furthering optimize performance.

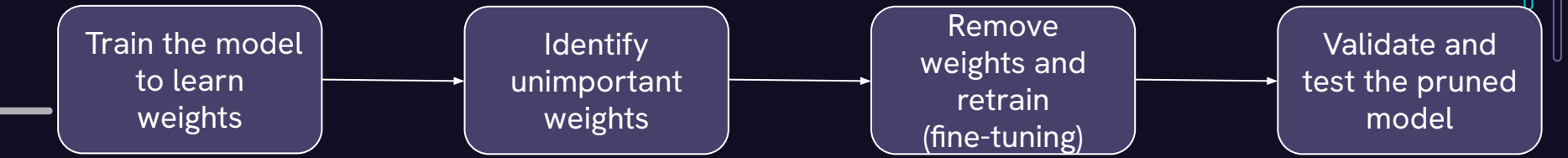
Pruning

Removes less important weights or connections in a neural network to reduce complexity while retaining accuracy.

Why?

1. Reduces model size.
2. Speeds up inference.
3. Saves memory and energy.

How Pruning Works?



Quantization

Reduces the precision of model weights and activations from 32-bit floating-point to lower precision, such as 8-bit integers.

Why?

1. Reduces model size (up to 4x smaller).
2. Accelerates inference on specialized hardware (e.g., GPUs, TPUs, and FPGAs).
3. Lowers memory and energy requirements.

Other Methods (Path 1)

- **Mixed Precision Training**:
  - Uses 16-bit floating-point arithmetic on compatible GPUs for faster training.
- **Accelerated Linear Algebra (XLA)**:
  - Just-In-Time (JIT) compilation to optimize computational graphs.
- **Efficient Data Pipelines**:
  - Using **`tf.data`** for optimized data handling.
- **Early Stopping**:
  - Halts training if validation loss stagnates to prevent overfitting and save computation.
- **All implemented via Tensorflow and Tensorflow Model Optimization Toolkit (TF-MOT)**

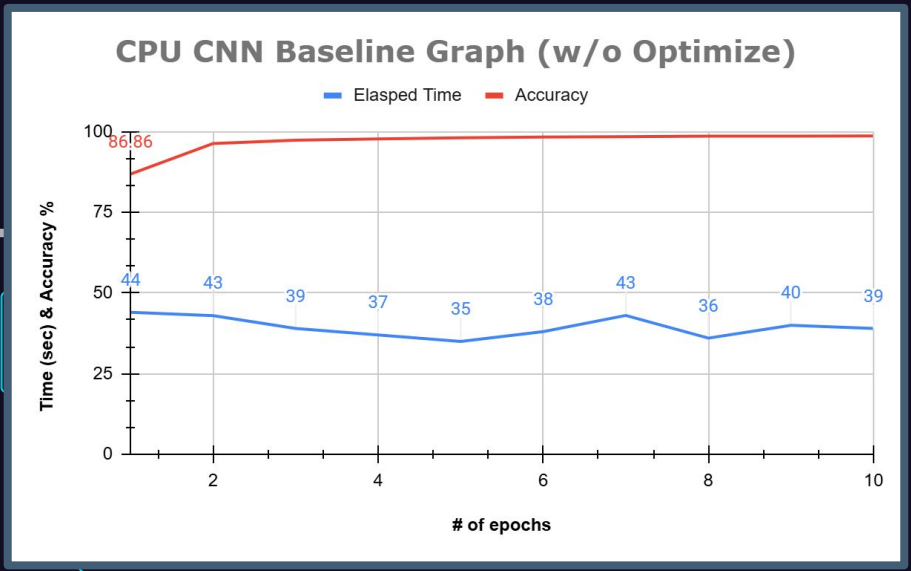
Path 2 Methods

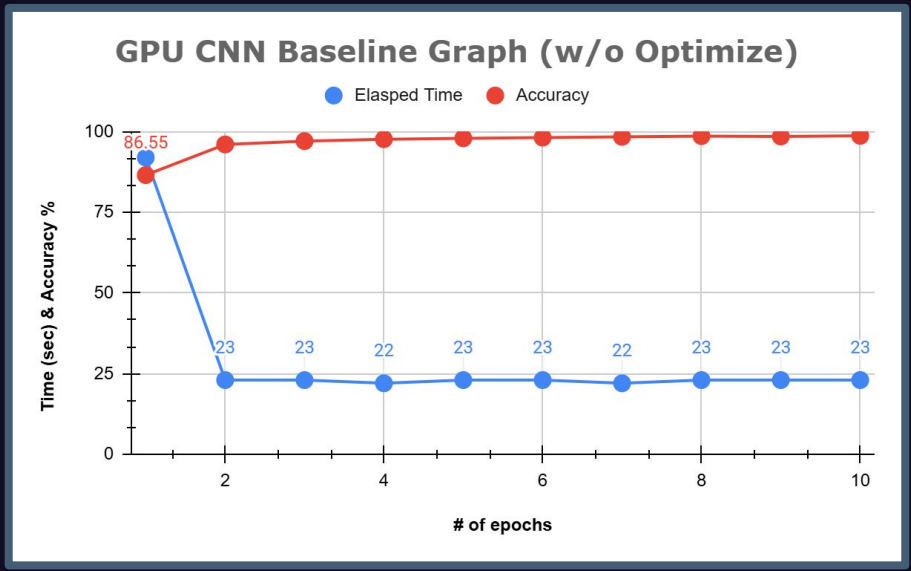
- **Tuning CPU Configuration**:
  - **CPU Frequency Scaling**: Setting CPU to “performance” mode (~3.7 GHz) vs. “powersave.”
- **Linux Kernel Parameter Tweaking**: Further optimization performance
- **Workload Scheduling**: Leveraging cgroups.
- **Thermal Management**: Using a PID-controlled fan script to cap GPU temperatures at 85 degrees C.

Results

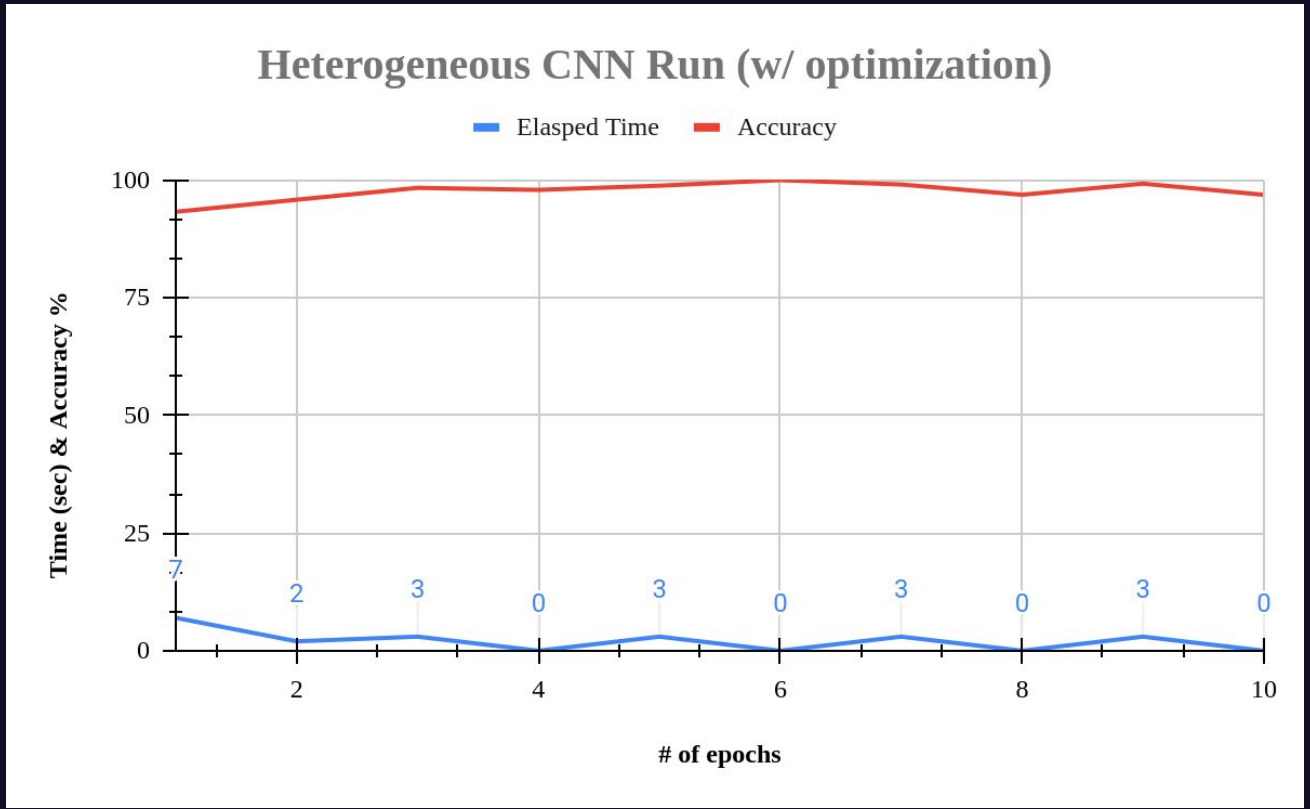
Phase 1: Data Collection & Baseline Testing

Speed (Phase 1)

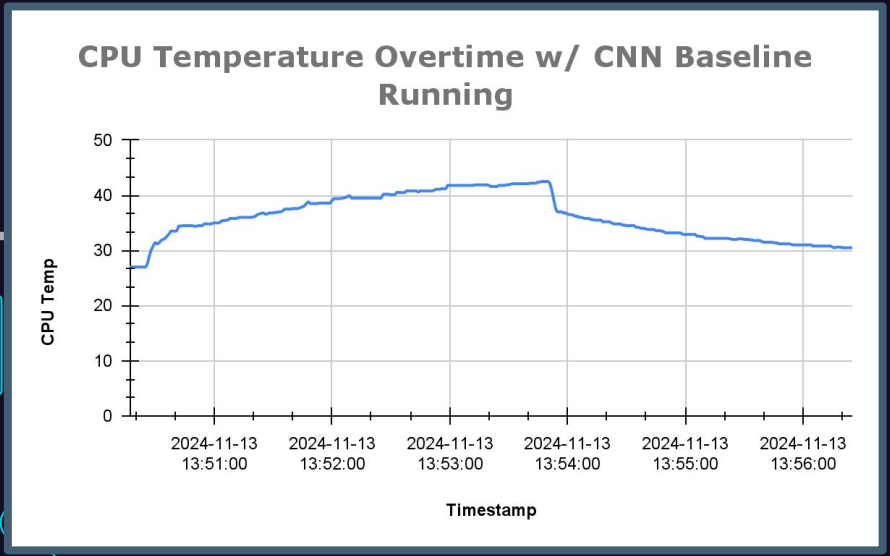


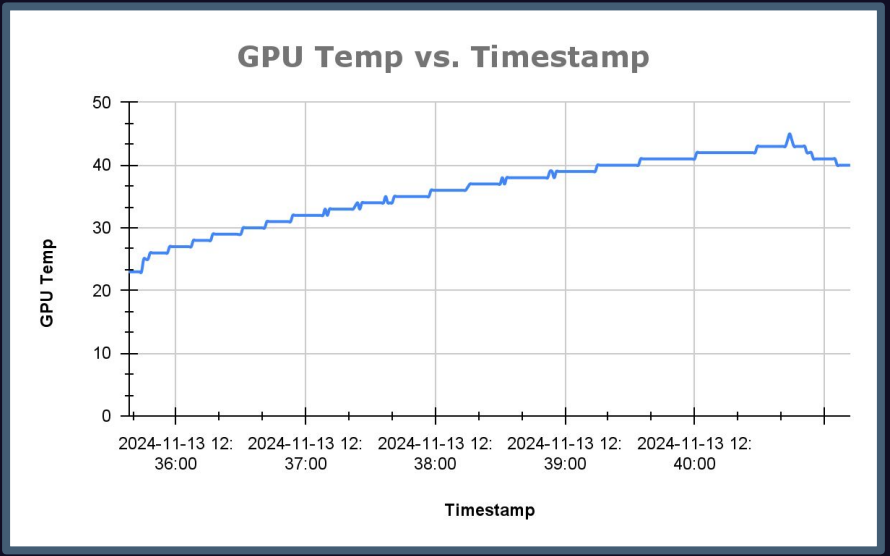


Speed (Phase 2)

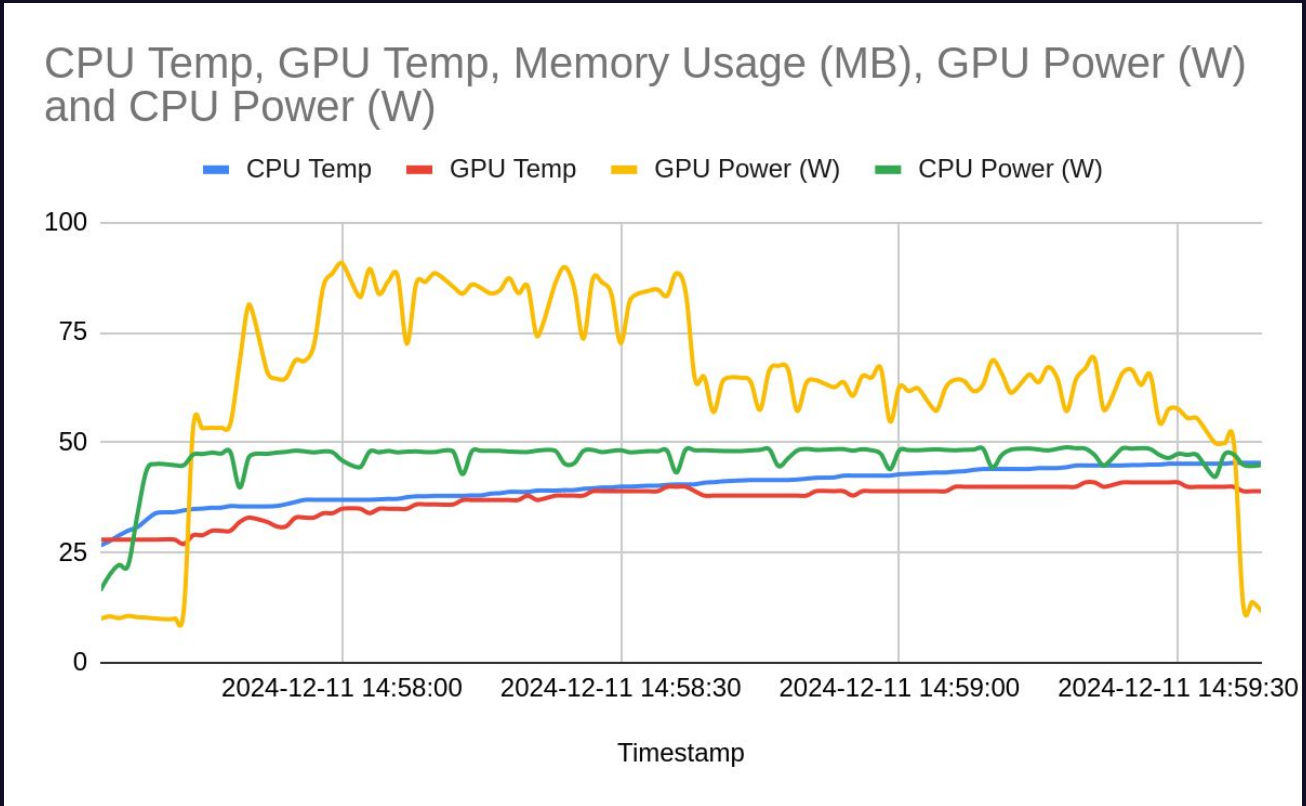


Thermal Temperature

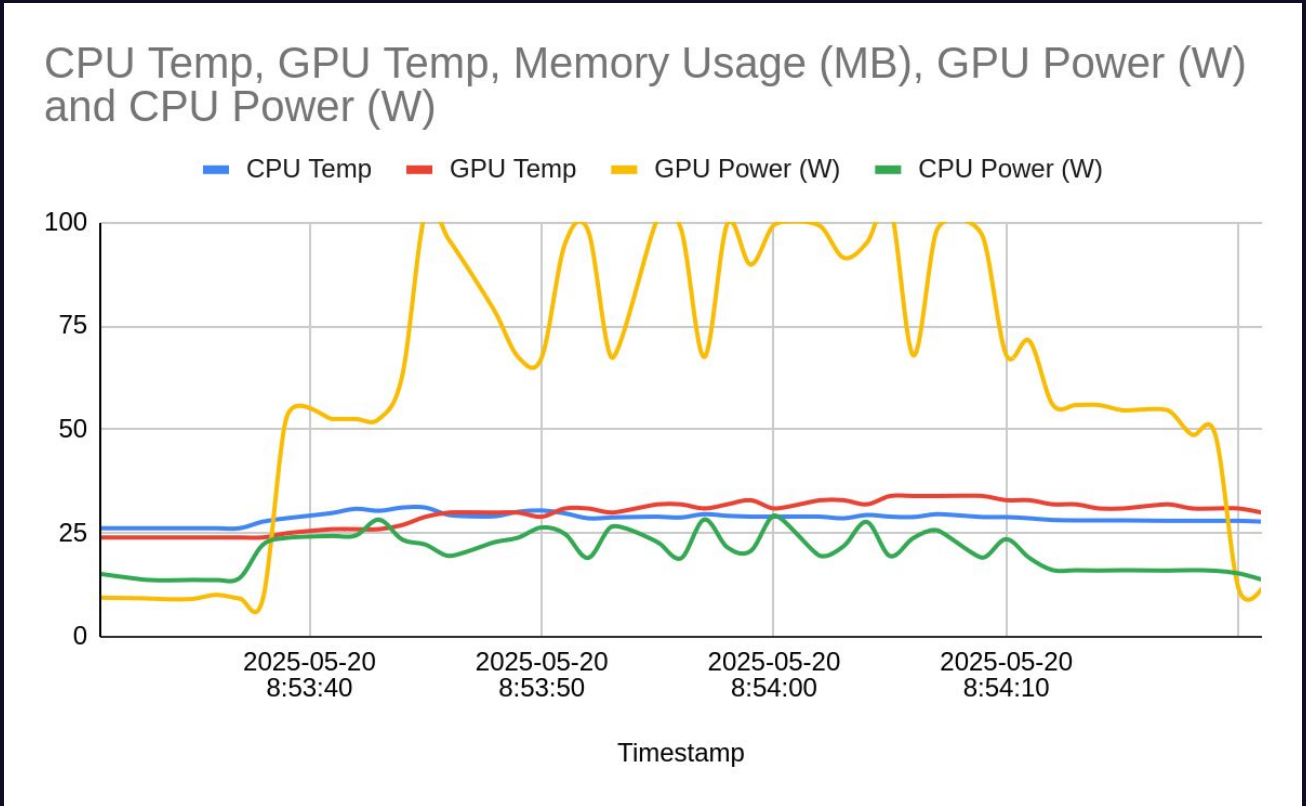




Full Graph


CPU Temp, GPU Temp, Memory Usage (MB), GPU Power (W) and CPU Power (W)
CPU Temp
GPU Temp
GPU Power (W)
CPU Power (W)
100
75
50
25
0
2024-12-11 14:58:00
2024-12-11 14:58:30
2024-12-11 14:59:00
2024-12-11 14:59:30
Timestamp

Full Graph (Phase 2)



Discussion

Key Findings

- Software optimization (pruning, quantization) significantly reduced model size and training time
- Hardware tuning enhances thermal stability and efficiency.
- Framework scalable to larger models.
- **Limitations**:
  - Mixed precision requires compatible GPUs; high sparsity needs fine-tuning.
- **Future Work**:
  - Dynamic scheduling, quantization-aware training, multi-node systems.

Key Findings (cont.)

- **Achievements**:
  - **Reduction** in CNN training time
  - **Model size compression**
  - **Thermal improvement**
  - **Blueprint for efficient AI processor design**
- **Impact**:
  - Scalable framework for data centers and edge computing, advancing sustainable AI deployment.

Acknowledgements

Dr. Shane Torbert for mentorship

The TJHSST Computer Systems Lab and Sysadmins for resources

Peers for feedback and support

Questions ?

Project Website

https://heliothon.github.io/heliothon-website