Enhancing Human Activity Recognition with Hardware Acceleration

Louie Cai louie@ucsd.edu
Jeffery Lai j8lai@ucsd.edu
Lian Gan lgan@ucsd.edu
Christian Dermanualian cdermanuelian@ucsd.edu

Rajesh Gupta rgupta@ucsd.edu

Abstract

In this project, we aim to enhance the performance of Human Activity Recognition (HAR) by shifting the focus from traditional software optimization to specialized hardware acceleration. Recognizing the limitations of General-Purpose GPUs in efficiently handling the unique computational demands of HAR, our approach involves a thorough analysis of our HAR model's baseline performance on standard GPU setups, followed by exploring specific hardware acceleration techniques. This targeted strategy is anticipated to significantly improve efficiency, reduce latency, and address bottlenecks, thereby optimizing HAR systems for more effective real-world applications

Code: https://github.com/louiecai/DSC-180B-Hardware-Acceleration

1 Introduction

The field of computing is rapidly evolving with the widespread integration of AI and ML into everyday applications. These programs and models collect and compute data on an unprecedented scale, necessitating their own specialized hardware. As a result, there has been a pivotal shift from software-based acceleration to hardware acceleration, as a CPU simply lacks the computational and storage resources to efficiently train and run these complex models.

2 Methods

2.1 Experimental Design

Our study aims to investigate the performance speedups of various Human Activity Recognition (HAR) neural networks across different hardware configurations, focusing on the computational efficiency and scalability of these models.

We implement a range of different experiments for our model executions.

- Experiment 1 - Our first was a simple linear scaling of CPU specifications. We take four environments, the first one having 1 vCPU (virtual cores) and 2 GiB of CPU memory. Both features scale linearly with a factor of 2. So, the final environment of this experiment has 8 vCPU and 16 GiB of memory.
- Experiment 2 - Our second experiment tests the impact of CPU memory on our executions. We use 3 environments each with 4 vCPUs. However, we scale up the CPU memory linearly with a factor of 2. The environments have 8, 16, and 32 GiB of memory in that order.
- Experiment 3 - Our third experiment is similar to Experiment 2, but with scaling vCPUs instead of memory. Each 3 environments have 16 GiB of memory, with linearly scaling vCPUs by a factor of 2. (Starting with 2 vCPU and ending with 8)
- Experiment 4 - Our fourth experiment introduces Graphics Processing Units. Here, we have 10 environments. Half of them have the Nvidia 2080ti, and the other half have Nvidia a5000 graphics cards. In each half, we keep the 5 environments at 32 GiB of memory and again scale the vCPUs. The vCPU scale for each half looks like [1,2,4,8,12].
- Large Sample -

A detailed tabular format of the environment can be found here. (Note that some environments on this list were not used (env11-env16))

We have selected this range of hardware configurations to match practical use cases of HAR applications. For example, a single-core CPU closely resembles the processing capability of wearable HAR devices like smartwatches or rings. Whereas larger devices like an 8-core CPU are meant to explore the impact of multi-threading on performance. The memory configurations will range from 4GB, suitable for low-end devices, to 32GB for high-end computing environments, ensuring a broad spectrum of data handling capabilities. Our

GPU trials are meant to give insights to how small-scale GPU with tensor cores can benefit certain CPU configurations.

The selection of HAR neural networks for our study is based on architectural diversity and computational demand. We plan to evaluate traditional Deep Neural Networks (DNNs), such as Multi-Layer Perceptrons (MLPs), Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Transformers. This variety will allow us to explore how different computational requirements affect speedups across our hardware spectrum.

No preprocessing or data augmentation techniques will be applied to ensure the data used in our experiments closely mimics real-world sensor data, maintaining the practical relevance of our findings.

2.2 Hardware Specifications and Software Environment

To ensure a controlled and consistent experimental environment, we utilize Docker containers configured with Ubuntu 22.04, managed through Conda environments. This setup features a standardized software stack, including the same versions of PyTorch for model training and evaluation, ensuring comparability across tests. Each container is designed to mount a local code repository, allowing for real-time code adjustments from the local machine without compromising the experimental environment's integrity.

You can find the Dockerfile here.

2.3 Models and Datasets

The study will utilize datasets derived from real sensors embedded in HAR devices, featuring data from Inertial Measurement Units (IMUs) among other sensors. These datasets will be selected to represent a wide range of activities, ensuring the models are tested against diverse and challenging scenarios.

2.4 Performance Metrics

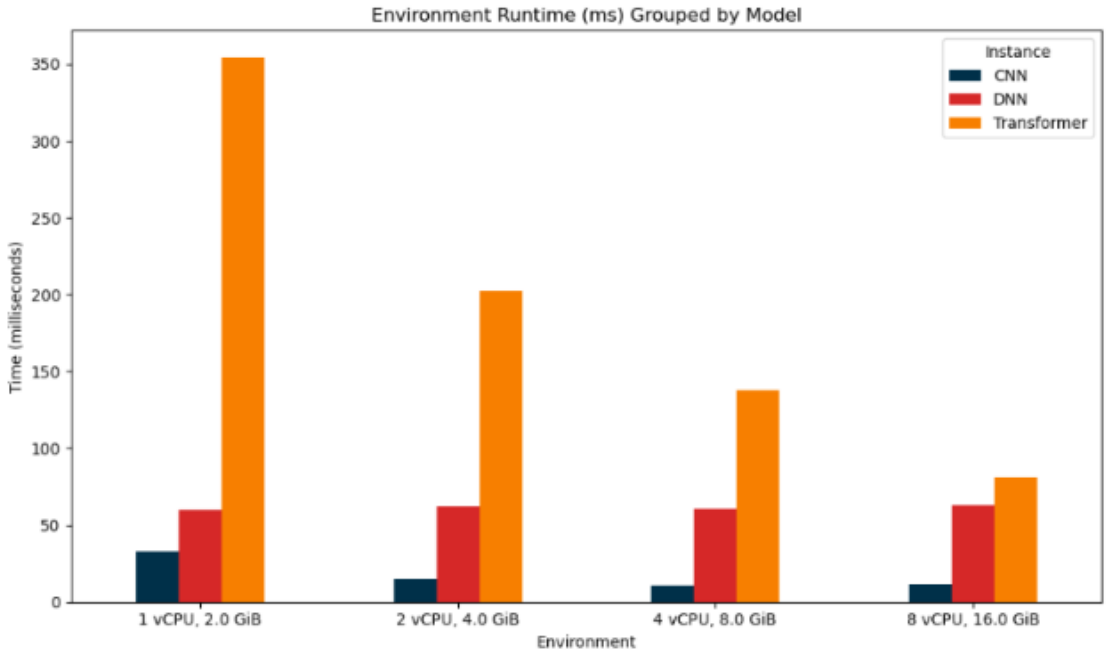
Our primary focus will be on measuring the speedups achieved through different hardware configurations, examining how the computational power and memory availability impact the training and inference times of each model. To ensure comparability across hardware configurations, we will utilize the same Docker container across all tests, with all experiments conducted on x86 CPUs. This approach will provide a consistent baseline for performance measurement, isolating the hardware's impact on speedup. Detailed performance metrics, including training time, inference time, and other relevant measures, will be defined and analyzed as part of our ongoing research.

2.5 Conclusion

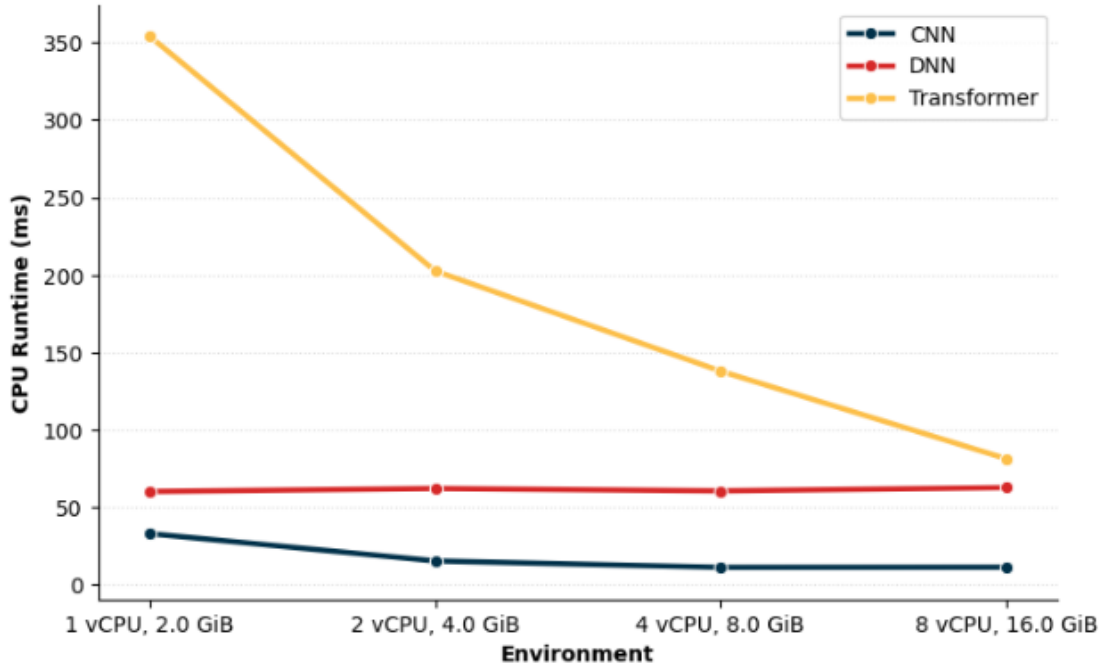
This section outlines the methodological framework of our study, emphasizing the experimental design, hardware and software configurations, and the selection criteria for HAR models and datasets. As our work progresses, we will update the specifics related to model evaluations, dataset characteristics, and detailed performance metrics to paint a comprehensive picture of HAR neural networks' efficiency across varying hardware platforms. Our approach underscores a forward-looking methodology, anticipating the completion of various components of our study.

3 Results

3.1 Experiment 1



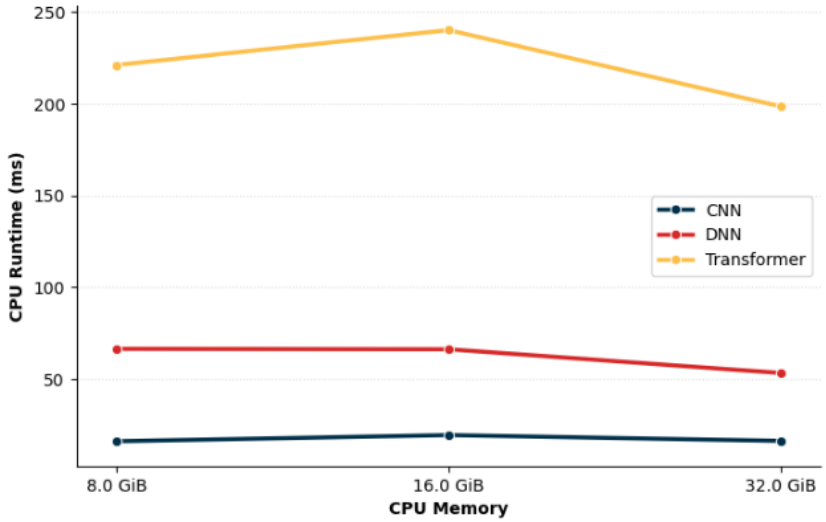
As we see above, it looks as though scaling up CPU specifications decreases overall runtime. We will take a deeper look below.



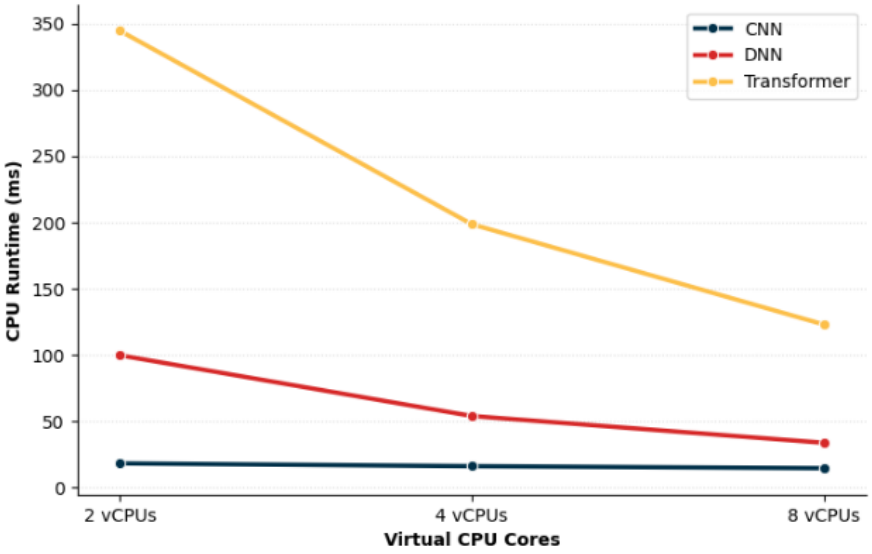
With the line graph, we see that we only gain performance for the Transformer model, while the CNN and DNN models get very little if any speedups. We will dive into why this happens with the following 2 experiments.

3.2 Experiments 2 & 3

Recall that Experiment 2 deals with constant vCPU cores with scaling CPU memory.



Experiment 3 is the inverse, with constant GPU Memory and scaling vCPU cores.



Above, we see which CPU feature matters. With scaling memory we see almost no performance gains for any of the models. In fact, the Transformer gets worse at one step.

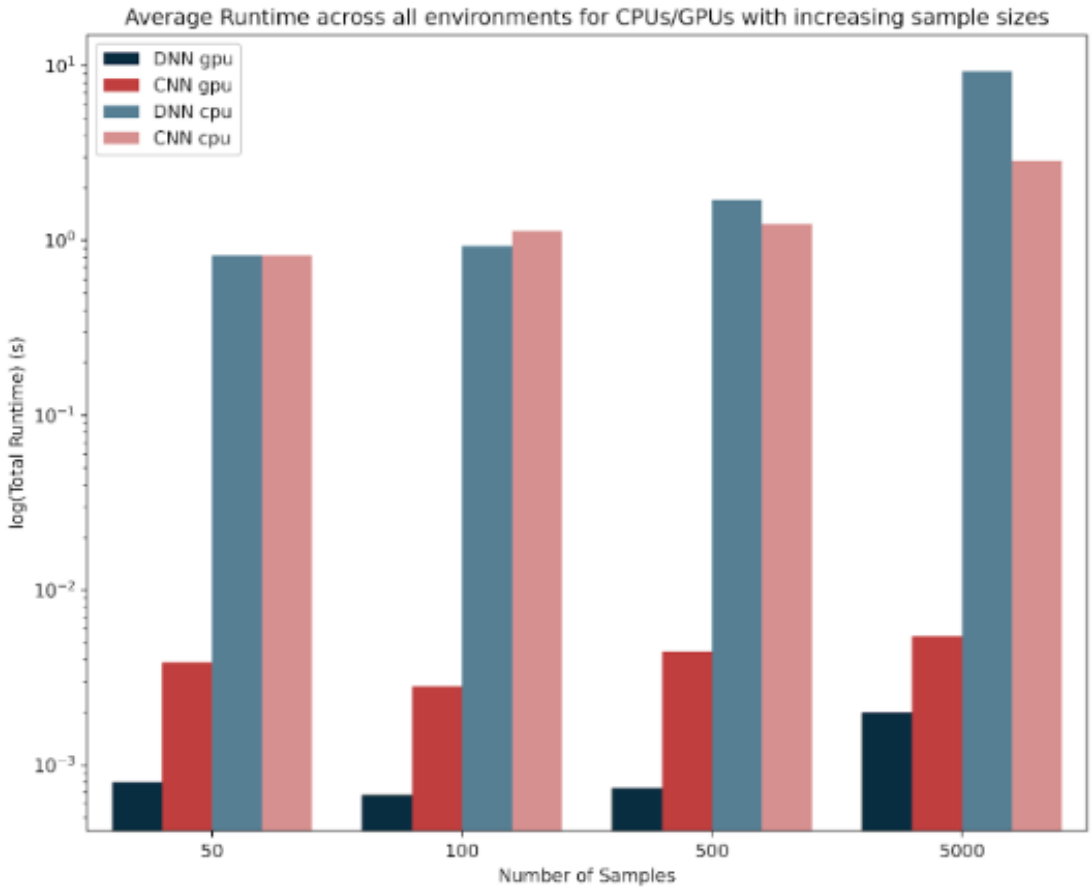
However, when scaling CPU cores we see that the Transformer and the DNN get progessively faster as the number of cores increases. The same cannot be said for the CNN, as it only marginally runs faster. The key takeway is that when it comes to CPUs, our model executions gain performance as they have access to more cores.

3.3 Experiment 4

Now we know that scaling the vCPU cores will increase performance. For the GPU experiment, we now can simply analyze the effect of one of our environments for each GPU. The most likely environment that will be used in a HAR application will involve smaller CPUs, so we will use the 1 vCPU and 32 GiB memory environments for each GPU Type. Recall that our GPU types are Nvidia 2080ti, and the Nvidia a5000.

After our data collection, we saw extreme performance gains for the CPU when we introduced both GPUs we saw an average of a 2000% speedup between both GPU types. This applies linearly when we scale the vCPU cores. The takeway is still that increased CPU cores increases performance, while adding a GPU adds a major contriubtion to any number of cores.

3.4 Experiment 5



As we increase the number of samples, the effect on CPU runtime is significantly greater than the increase of runtime on the GPUs. The effect is not noticeable on one sample and the GPU instances may take longer than their CPU counterpart since they take time to initialize. However, it is much more effective when running a large number of samples in parallel.

4 Impact & Next Steps

Our project has primarily executed networks on general-purpose CPUs and GPUs, tapping into their reliable computational power for Human Activity Recognition (HAR) system advancements. This foundational approach has enabled us to make significant strides in enhancing system efficiency and processing capabilities.

However, the potential of integrating Field Programmable Gate Arrays (FPGAs) into our framework presents an intriguing avenue for future exploration. Unlike CPUs and GPUs, FPGAs offer a customizable platform, allowing for the hardware to be precisely tailored to specific computational tasks. This flexibility suggests that FPGAs could dramatically accelerate our networks, potentially surpassing the performance gains currently achieved with traditional computing resources.

While we have yet to venture into FPGA-based execution, the theoretical advantages of such technology—ranging from increased processing speeds to greater energy efficiency—underscore its promise for HAR systems. FPGAs' capacity for high-speed data processing and adaptable architecture could unlock new levels of performance and customization.

As our project evolves, exploring the use of FPGAs may offer groundbreaking opportunities to further revolutionize HAR applications. This forward-looking perspective not only aligns with our commitment to innovation but also opens the door to harnessing the full potential of cutting-edge hardware acceleration techniques in the future.

5 Contributions

This section outlines the contributions of each team member to the project. The collaborative effort and individual responsibilities are detailed below, reflecting the diverse expertise and commitment of the team towards achieving the project goals.

Christian:

- Analayzed HAR Dataset to help generate new data
- Researched CPU and GPU hardware processes

Jeff:

- Set up AWS Account
- Uploaded and tested Docker image on AWS Server
- Researched optimal AWS servers to test

Francis:

- Cleaned data
- Finalized two models
- Setup profiling code

Louie:

- Designed the workflow/methods of this project
- Created the Docker image and container (documentation)
- Researched PyTorch profiler

The collective efforts of all team members have been pivotal in driving the project forward. Each contribution, whether in the form of research, development, analysis, or documentation, has played a crucial role in the project's progress and success.