Project Report: Performance comparisons between Resistive and static RAM for Machine Learning Models

Ramesh Sah
ramesh.sah@wsu.edu

Hong-Ying Lin
hong-ying.lin@wsu.edu

Oluwafemi Ajeigbe
oluwafemi.ajeigbe@wsu.edu

Team Name and the Project Category
Our team name is group 3 and we have chosen project category 3: Emerging computer architecture paradigms for machine learning, in-memory computing, deep neural networks, graph neural networks, etc.

INTRODUCTION

1.1 Research Problem

Machine learning systems are increasingly becoming an integral part of our everyday life [3]. Machine learning is now used in high power and large bandwidth systems such as supercomputers and cloud servers as well as in resource-constrained environments such as edge computing, and Internet-of-things (IoT). Furthermore, specialized architectures of machine learning algorithms are used for specific applications. For example, convolutional neural networks (CNN) and their variants are widely used in computer vision (CV) applications and spatiotemporal architectures such as Long Short Temporal Model (LSTM), and recurrent neural networks are used in natural language processing (NLP). Many non-Von Neuman architectures have been proposed to efficiently run these machine learning algorithms.

Training and using deep learning algorithms involves storing millions of parameters and executing predominantly multiply-accumulate (MAC) operations. To facilitate these, new computer architectures such as compute-in-memory (CIM) have been proposed. In the CIM paradigm, MAC operations are done in memory or near memory so that the memory bandwidth bottleneck is removed. While in-memory computing has been studied on different memory devices such as static RAM (SRAM), resistive RAM (ReRAM), NVMs, and dynamic RAM (DRAMs) there is yet to work comparing the efficacies of these memory technologies for running machine learning algorithms. Since each memory architecture has its benefits and challenges our goal is to study the heterogeneity aspect of these memory architectures. We want to propose the best heterogenous CIM configuration for deep learning hardware using SRAM and ReRAM memory device types.

1.2 Motivation

The success of machine learning algorithms in many previously unsolved problems begs the question of accelerating machine learning algorithms using specialized techniques. Moreover, heterogenous CIM architectures have not been explored yet even though both ReRAMS and SRAMs offer significant benefits. We understand that write energy is less on SRAMs than ReRAMs and ReRAMs are denser than SRAMs. We want to explore the trade-off offered by SRAMs and ReRAMS for running deep learning models on a layer-level scale. We hope our analysis will shed light on efficient hardware design for running/accelerating machine learning algorithms. Memory paradigms that are efficient in terms of power and memory and faster in execution time will open up the usage of state-of-the-art machine learning algorithms in a broad set of systems and for a broad set of applications [2].

1.3 Related Work

SRAM and ReRAM technologies have been studied and compared for their read-write energies and have been a consideration in the past [1]. We aim to use these studies as a base to achieve our objectives in this project. In [4] the authors evaluated the performance of partitioning a 512 X 512 weight matrix into the SRAM and ReRAM-based accelerators. With more partitioning and finer granularity of the array architecture, the read/write latency and the dynamic read/write energy will be decreased. This was due to the increased computation parallelism at the expense of a larger area and leakage power. However, the ReRAM accelerator did not improve the read latency and read energy beyond a certain point due to the overhead of multiple intermediate stages of adder and registers [4].

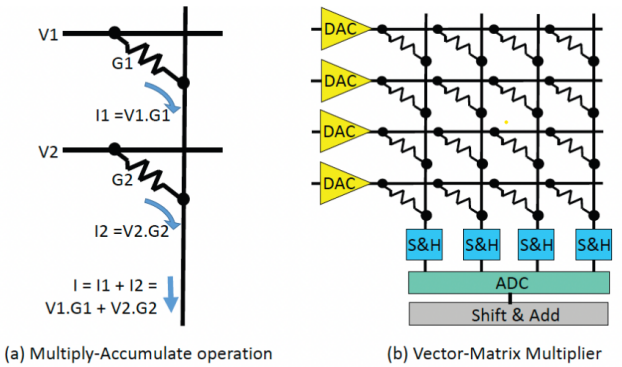

Fig 1. (a) A bit line is used to perform an analog sum of products operation, (b) A memristor crossbar is used as a vector-matrix multiplier.

Figure 1 shows the multiply-accumulate (MAC) operation in ReRAM. Current *I* is the dot product between the

voltage vector V at each row and the conductance vector G in each column.

$$I = V * G$$

A typical 6T SRAM cell is shown in Figure 2. The resultant discharge voltage on the BLB represents a one-bit multiplication on the data stored in the SRAM cell (W) with the multi-bit input voltage (Vin) applied as an analog voltage on the word line.

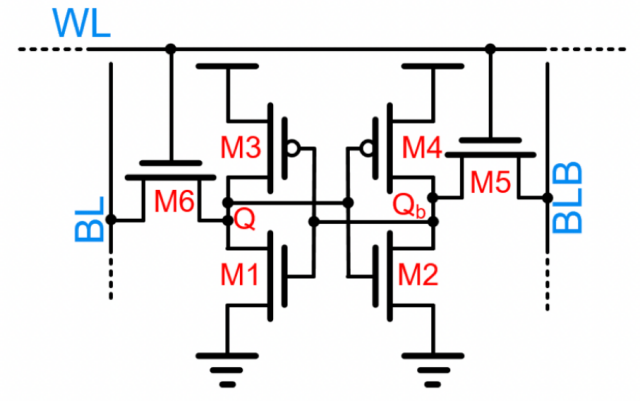

Fig 2. 6T SRAM Cell

Contributions: We will analyze the efficacies of SRAM and ReRAM for running deep machine-learning models. We compare the performance of SRAM and ReRAM in terms of reading speed and power requirements with inter-model and intra-model analysis. In the inter-model, we will compare the performance parameters for different types of deep learning models. In the intra-model, we will analyze the model on a per-layer basis.

Ramesh: Literature review, understanding neurosim tools, machine learning models, and running experiments.

Oluwafemi: Literature review, setting up the experiment environment, and running experiments.

Hong-Ying: Literature review, setting up the experiment environment, and concluding results.

PROPOSED APPROACH

1.1 Outline of the Proposed Approach

We propose to study the efficacies of SRAM and ReRAm memory devices for accelerating deep learning models. We intend to do a design space exploratory analysis on these memory device types using the Neurosim simulator.

1.2 Execution Plan

The research tasks described in Section 1.1 will be executed as follows. Ramesh will do the literature survey and determine the tools and models used in the analysis part of the project. Oluwafemi will set up the Neurosim simulator and run the experiments. Finally, Hommy will analyze the results and create graphs and figures to highlight the findings.

EXPERIMENTAL EVALUATION

1.1 Methodology

We will use the *Neurosim* simulator in our analysis. Neurosim [5] is developed in C++ and wrapped in PyTorch to emulate deep neural network performance based on near-memory or in-memory computing architectures. Neurosim supports various device technologies such as SRAM, NVM, and ReRAM; it allows hierarchical structures at the device, circuit, chip, and algorithm levels.

We will begin by converting a deep neural network into a comma-separated value (CSV) file containing information about the model architecture such as feature height/width, kernel size, and layers. We have selected VGG8 and DenseNet 40 computer vision models for analysis.

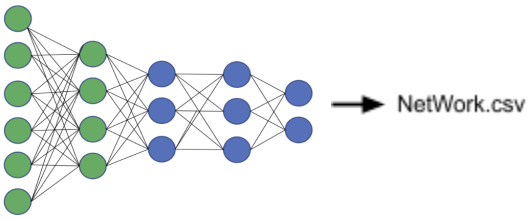

Fig 3. Converting Deep Neural Network

After obtaining the NetWork.csv file we modify the hardware parameters in the *Param.cpp* file of the NeuroSim. We can change parameters such as technology node, **device type (SRAM / eNVM / ReRAM / FeFET)**, operation mode, synaptic subarray side, synaptic device prevision, mapping method, activation type, and clock frequency. After making the hardware changes, the NeuroSim library is compiled to generate binaries using *make*. Next, we simulate the machine learning model using the Python wrapper of the NeuroSim as shown in figure 4.

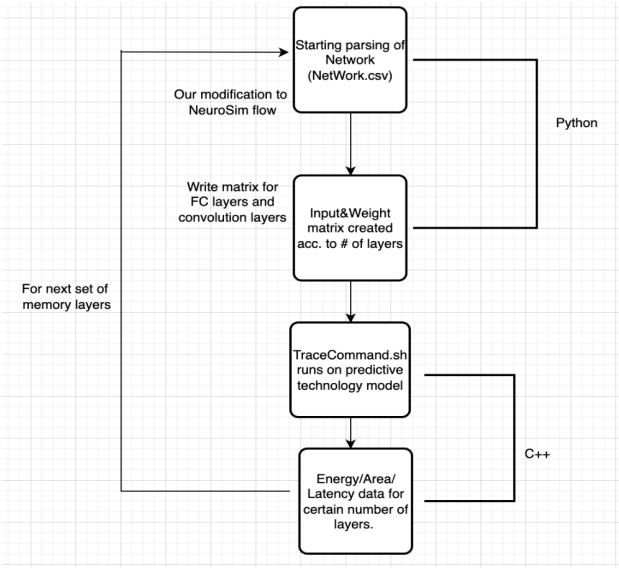

Fig 4. NeuroSim Workflow

1.2 Results

Figure 5 (a) to (e) shows the results from our analysis with models VGG8 and DenseNet40. Our goal is to understand the advantages of ReRAM and SRAM for accelerating deep neural networks. From figure 5(a), the chart shows that RRAM has less latency compared to SRAM, and from figure 5(b), the chart shows that RRAM cost 40% less dynamic energy than SRAM in model VGG8, but no significant difference in model DenseNet40.

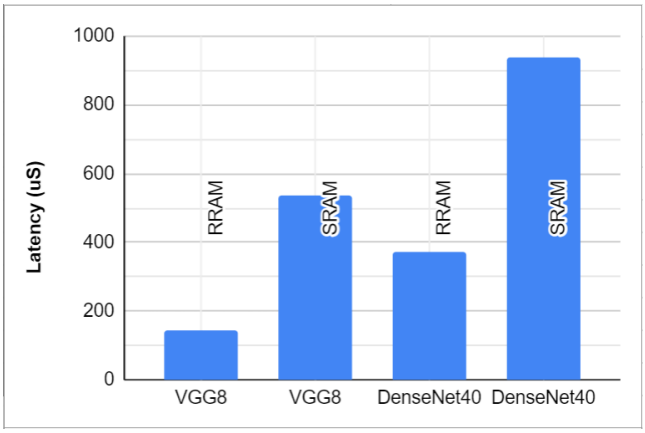

Fig 4. (a) Latency vs CIM Technology

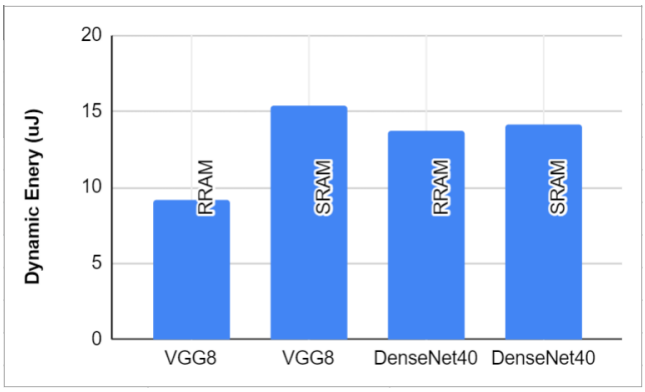

Fig 4. (b) Dynamic Energy vs CIM Technology

From figure 5(c), the chart shows that RRAM is one time more efficient in terms of energy than SRAM and from figure 5(d), SRAM possesses more frame rate than SRAM. Both figures 5(a) and 5(d) indicate that RRAM is faster than SRAM. From figure 5(e), both RRAM and SRAM are roughly the same in utilization.

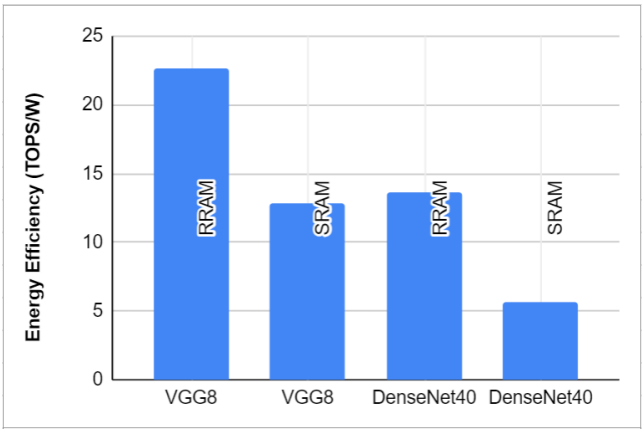

Fig 4. (c) Energy Efficiency vs CIM Technology

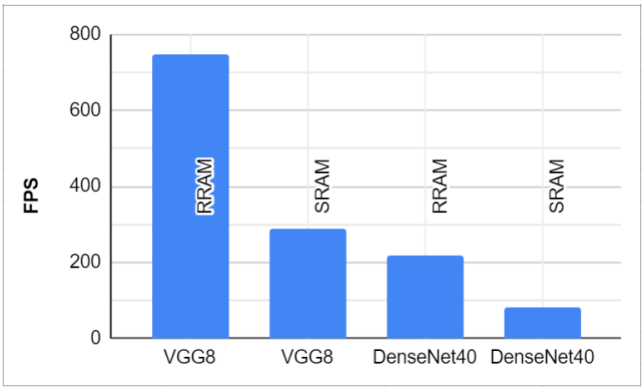

Fig 4. (d) FPS vs CIM Technology

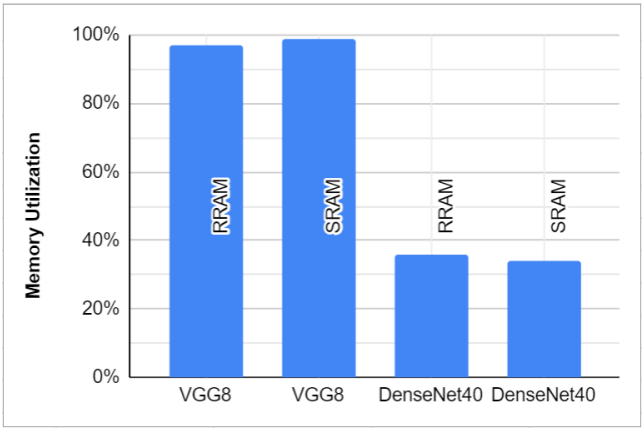

Fig 4. (e) Memory Utilization vs CIM Technology

Conclusions and Future work

Our analysis showed the read performance comparisons for SRAM and ReRAM compute-in-memory technologies. We found ReRAM to be energy efficient and faster for read operations. Faster computation in ReRAM means the model processes more input images and consequently

models running on ReRAM had higher frames-per-second. Memory utilization was similar for both ReRAM and SRAM operations. Unfortunately, we couldn't extend our analysis to measure writing performance, and we hope to complete this in a future project. We also aim to extend our analysis to other types of machine learning models such as natural language processing and graph learning. The current version of NeuroSim only supports quantized and floating point versions of Convolutional and Fully-Connected layers and consequently greatly limits the coverage of our analysis.

Acknowledgment

We would like to thank Dr. Bhat for the opportunity to conduct this research project and for his support in various issues.

REFERENCES

[1] Shafiee, Ali, et al. "ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars." *ACM SIGARCH Computer Architecture News* 44.3 (2016): 14-26.

[2] Chen, Tianshi, et al. "Diannao: A small-footprint high-throughput accelerator for ubiquitous machine-learning." *ACM SIGARCH Computer Architecture News* 42.1 (2014): 269-284.

[3] Larochelle, Hugo, et al. "An empirical evaluation of deep architectures on problems with many factors of variation." *Proceedings of the 24th international conference on Machine learning*. 2007.

[4] Chen, Pai-Yu, and Shimeng Yu. "Partition SRAM and RRAM based synaptic arrays for neuro-inspired computing." *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2016.

[5] Peng, Xiaochen, et al. "DNN+ NeuroSim: An end-to-end benchmarking framework for compute-in-memory accelerators with versatile device technologies." *2019 IEEE international electron devices meeting (IEDM)*. IEEE, 2019.