EE746 Project Stage 1
Resistive Processing Unit for Analog Memories - FeRAMs

Koustav Jana (170070051), Mihir Kavishwar (17D070004), Prashant Kurrey (17D070057)

1 Introduction and Motivation

Deep Neural Networks (DNNs) have become ubiquitous in several Machine Learning (ML) applications such as image classification, speech recognition, natural language processing, etc. As the ML models are becoming increasingly complex, the required computational resources for both training and testing are also growing. In microprocessors based on Von-Neumann architecture, the tasks of memory access and computation are performed separately. Therefore, large ML models take significant time for training even on the best Graphic Processing Units (GPUs). Moreover, such GPUs are also power hungry and cannot be used for Edge applications where energy efficieny is very important.

Resistive Processing Units (RPUs), which are based on the principle of In-Memory computing, offer a promising solution for both the aforementioned problems. RPUs can perform tasks such as Matrix-Vector-Multiplication (MVM) and vector-vector outer product in a highly parallel fashion ($\mathcal{O}(1)$ time complexity) and with high energy efficiency. An RPU is made up of resistive crossbars, which have Non-Volatile Memory (NVM) element at every intersection to store a weight. Previous works have used varies NVMs such as Resistive RAM, Phase Change Memory and Charge Trap Flash. In this project we want to explore if FeRAM can be potentially used as a NVM in RPUs.

2 Literature Survey

We studied four different papers, two of which are focused on RPUs (Gokmen et al., 2016 [2], Bhatt et al., 2020 [3]) while the other two investigated Ferroelectric memories (Fan et al., 2016 [1], Mulaosmanovic et al., 2020 [4]).

The work presented in [2] proposes and analyzes the concept of RPU devices that can simultaneously store and process weights and are potentially scalable to billions of nodes with foundry CMOS technologies. The paper discusses the tolerance of the training algorithm to various RPU device and system parameters as well as to technological imperfections and different sources of noise. The work also presents an analysis of various system designs based on the RPU array concept that can potentially provide many orders of magnitude acceleration of DNN training while significantly decreasing required power and computer hardware resources.

In [3], the authors present a charge-trap-flash (CTF) device that can act as a cross-point device in the RPU framework for accelerating DNNs while maintaining software-level accuracy. The paper discusses how forward pass as well as backpropogation can be implemented for a DNN using the RPU architecture using stochastic pulse trains to perform analog multiplication. Experimental results are presented which show a high number of conductance levels and approximately linear updates by choosing appropriate pulse width and voltage for weight update. Simulation results of this architecture with standard datasets like MNIST, CIFAR-10 and CIFAR-100 show good accuracy levels. In addition to supervised learning problems, the work also showed successful training of a reinforcement learning agent on the Mountain Car environment.

In [4] experimentally investigated Accumulative Switching and One-Shot Switching in large and small-area Fe-FETs. It showed that both switching types obey the universal time–voltage dependence, irrespective of the device size, pulse polarity, or time intervals between the excitation pulses. The paper [1] reviews the most appealing topics about ferroelectric HfO_2- based materials including origins of ferroelectricity, advantageous material properties, and current and potential applications in FeRAM.

3 Method and Results

We implemented an RPU in MATLAB for an image classification task on the MNIST dataset. The NN model was same as that in [3] - a fully connected network with two hidden layers consisting of 256 and 128 neurons respectively. ReLU activation was used in the hidden layers while softmax activation was used for the output layer. Due to high simulation time on our machines, especially when using longer stochastic pulse trains, we could only train the network for 1 epoch. The model hyperparameters and simulation results are discussed on the next page.

4 Next Stage Plan

1. Add gaussian noise with varying (σ, μ) in the model and re-run the experiment
2. Use the V_T data for Ferroelectric device, which is inherently stochastic
3. Figure out a way to perform multiplication operation using deterministic pulses by exploiting inherent stochasticity of Ferroelectric crossbars.

5 References

[1] Zhen Fan, Jingsheng Chen, and John Wang. "Ferroelectric HfO 2 -based materials for next-generation ferroelectric memories". In: *Journal of Advanced Dielectrics* 6 (May 2016), p. 1630003. DOI: 10.1142/S2010135X16300036.

[2] Tayfun Gokmen and Yurii Vlasov. "Acceleration of Deep Neural Network Training with Resistive Cross-Point Devices: Design Considerations". In: *Frontiers in Neuroscience* 10 (2016), p. 333. ISSN: 1662-453X. DOI: 10.3389/fnins.2016.00333. URL: https://www.frontiersin.org/article/10.3389/fnins.2016.00333.

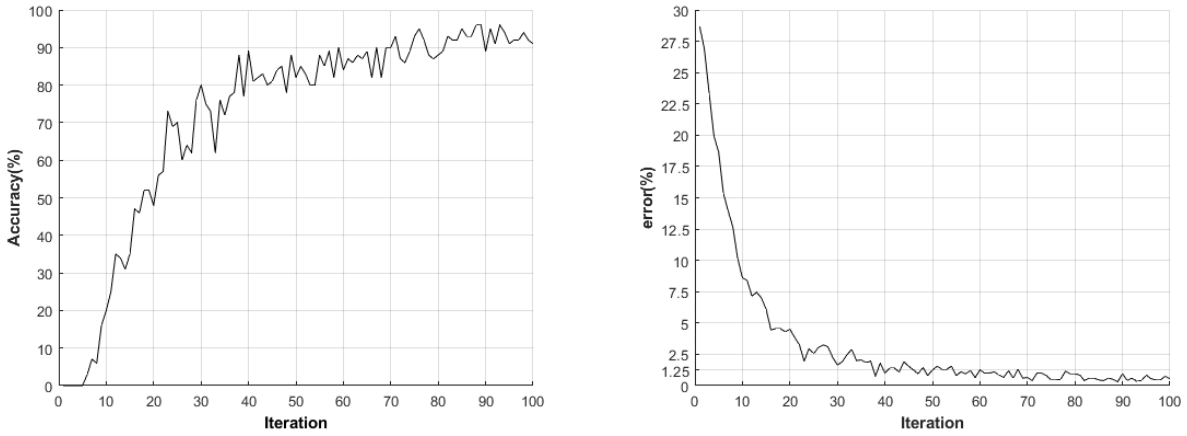
[3] Varun Bhatt et al. "Software-Level Accuracy Using Stochastic Computing With Charge-Trap-Flash Based Weight Matrix". In: *2020 International Joint Conference on Neural Networks (IJCNN)*. 2020, pp. 1–8. DOI: 10.1109/IJCNN48605.2020.9206631.

[4] Halid Mulaosmanovic et al. "Investigation of Accumulative Switching in Ferroelectric FETs: Enabling Universal Modeling of the Switching Behavior". In: *IEEE Transactions on Electron Devices* 67.12 (2020), pp. 5804–5809. DOI: 10.1109/TED.2020.3031249.

6 Figures and Tables

Hyperparameter	Value
Update step size (α)	0.01
Weight scaling factor (k)	600α
Initial device conductance $(g_{1,0}, g_{2,0})$	$\sim \mathcal{U}(0,1)$
Pulse train length PL	10
Input scaling factor C	$\frac{\alpha}{PL.\Delta^{+}g(c).k}$

Table 1: Hyperparameters



(a) Training Accuracy vs Number of Iterations

(b) Error vs Number of Iterations

Figure 1: MATLAB simulation results for RPU based of MNIST classifier. Training accuracy upto 92% was achieved in 1 epoch.