Adversarial Attack Mitigation Approaches Using RRAM-Neuromorphic Architectures

Siddharth Barve
University of Cincinnati
Cincinnati, Ohio, USA

Sanket Shukla
George Mason University
FairFax, Virginia, USA

Sai Manoj Pudukotai Dinakarrao
George Mason University
FairFax, Virginia, USA

Rashmi Jha
University of Cincinnati
Cincinnati, Ohio, USA

ABSTRACT

The rising trend and advancements in machine learning has resulted into its numerous applications in the field of computer vision, pattern recognition to providing security to hardware devices. Eventhough the proven achievements showcased by advancement in machine learning, one can exploit the vulnerabilities in those techniques by feeding adversaries. Adversarial samples are generated by well crafting and adding perturbations to the normal input samples. There exists majority of the software based adversarial attacks and defenses. In this paper, we demonstrate the effects of adversarial attacks on a reconfigurable RRAM-neuromorphic architecture with different learning algorithms and device characteristics. We also propose an integrated solution for mitigating the effects of the adversarial attack using the reconfigurable RRAM architecture.

CCS CONCEPTS

• **Hardware → Hardware reliability**; • **Security and privacy → Malware and its mitigation**; • **Computer systems organization → Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

KEYWORDS

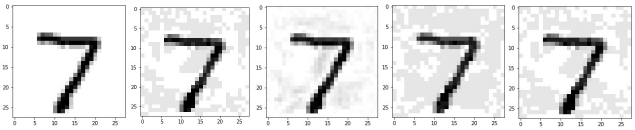
datasets, neural networks, gaze detection, text tagging

ACM Reference Format:
Siddharth Barve, Sanket Shukla, Sai Manoj Pudukotai Dinakarrao, and Rashmi Jha. 2021. Adversarial Attack Mitigation Approaches Using RRAM-Neuromorphic Architectures. In *Proceedings of the Great Lakes Symposium on VLSI 2021 (GLSVLSI '21), June 22–25, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/XXXXXX.XXXXXX

1 INTRODUCTION

Machine learning is an emerging technique and is extensively used in various fields like computer-vision, pattern-recognition, natural language processing, computer security etc. where the massive volume of data is generated regularly. Among several machine learning classifiers, the neural network class especially deep neural networks (DNNs) and convolutional neural networks (CNNs) have tremendously transformed the capabilities and computational power of the computer systems. Some of the machine learning applications in the aforementioned fields includes self-driving cars [26]. Advancements and progress in the field of computer-vision has anticipated the development of self-driving cars without any human intervention [7]. Similarly, machine learning has shown promising results to secure computer systems against malware and stealthy malware using image recognition [25] and pattern recognition [24] techniques. Despite the benefits and results showcased by advancements in machine learning, the existing vulnerabilities tend to exploit by impacting the performance of the machine learning classifier.

Although the machine learning techniques tend to be robust to the noise, the exposed vulnerabilties has shown that the output of machine learning classifier can be easily manipulated by crafting perturbations to the input data [6, 20, 27]. The data generated by crafting perturbations is generally known as Adversarial samples. These adversarial samples are constructed by perturbing the input data in one or multiple cycles iteratively under certain constraints in order to escalate the classification error rate.



(a) Original digit (b) BIM (c) DF (d) FGSM (e) MIM

Figure 1: (a) Original MNIST digit "7"; (b) BIM generated adversarial image; (c) DF generated adversarial image; (d) FGSM generated adversarial image and (e) MIM generated adversarial image

Figure 1 illustrates a simple adversarial sample generated from the MNIST digit dataset [13] for digit '7'. The Figure 1(a) is the orgiginal image which is classified as 7 by the neural network classifier. The images in Figure 1(b), 1(c), 1(d), 1(e) are generated by the basic iterative method (BIM), Deepfool attack (DF), fast gradient sign method (FGSM), and momentum iterative method (MIM) respectively. It can observed from the Figure 1(a), 1(c) and 1(e) that the normal and adversarial samples look similar for human observation. It needs to be noted that the noise in Figure 1(c) and

GLSVLSI '21, June 22–25, 2021, Virtual Event, USA.

ACM ISBN 978-1-4503-8393-6/21/06…$15.00
https://doi.org/10.1145/XXXXXX.XXXXXX

1(e) can be increased or reduced by tuning the parameters of the attack. With the change in attack parameters, the classifier output and it's confidence will be modified. More details on generating the adversarial attacks are presented in Section 2.

The adversarial attacks can be broadly classified into two categories: (a) poisoning attacks and (b) evasion attacks. Poisoning attacks are attacks on the ML classifier during the training phase [2, 17, 18, 22], and the evasive attacks are targeted for inference stage of machine learning techniques. Poisoning attacks are appropriate for online environments because they focus on attacking the classifiers during the training phase. Therefore, in this work we focus on the evasive attacks, as many of the existing machine learning works are primarily offline learning based and are constrained by resources and the computational time requirements.

Rise in the types of adversarial attacks led to development of adversarial defense techniques. Some of the prominent software based adversarial defenses includes adversarial training [6, 27], defensive distillation [21] and MagNet [14]. Even though these adversarial defense show some robustness against adversarial samples, they also have major drawbacks and weaknesses. More details over the adversarial defenses is presented in Section 2.

As the aforementioned defense is developed based on software, researchers have started to shift the focus from software to hardware. There have been a new research trending which comprises of leveraging neuromorphic computing to provide a robust defense against the adversaries. In [1] authors demonstrate the advantage conferred by the non-idealities present in analog crossbars in terms of adversarial robustness. Authors in the paper [11] propose a neuromorphic approach based on sparse coding. These neuromorphic based techniques are designed specifically for a targeted attack such as one-pixel attack and FGSM attack. The interoperability of these defense against other pool of adversarial attacks still remains a concern.

In this work, we first provide an overview of evasive attacks on the ML classifiers. Further, we present different existing defense techniques for the adversarial attacks. As FGSM is one of the fastest evasive attacks, an in-depth discussion regarding the FGSM adversarial attack is provided. In this work, we look at initially introduced defense against adversarial samples, Adversarial training is one of the defense techniques introduced for adversarial attacks. Further, in this paper we explore the potency of adversarial attacks on a reconfigurable RRAM-based Neuromorphic Architecture. We also propose a method for mitigating adversarial attacks in deployed IoT devices combining precise software training algorithms and the reconfigurable RRAM-based Neuromorphic architecture.

2 BACKGROUND

Adversarial samples are generated by introducing crafted perturbations into the normal input data generated. This makes the adversarial data look similar to the normal input data, but still the machine learning model mispredicts the class with a high probability. These adversarial samples can be considered as an optical illusion for the ML classifiers. In this section, we present different techniques widely used for generating the adversarial samples, and review some of the popular defense techniques deployed.

2.1 Adversarial Attacks

Here we present an overview of some of the adversarial attacks that are effective against machine learning classifiers.

2.1.1 *Fast Gradient Sign Method (FGSM)*

. The most common technique to perform adversarial attack is to perturb the image with gradient of the loss with respect to the image or input. Then gradually increase the magnitude of the perturbation until the image is misclassified.

Fast Gradient Sign method (FGSM) [6] is one of the first known adversarial attacks. The complexity to generate FGSM attack is lower compared to other adversarial attacks, even against deep learning models. This technique features low complexity and fast implementation. Consider a ML classifier model with θ as the parameter, x being the input to the model, and y is the output for a given input x, and L(θ, x,y) be the cost function used to train the neural network. Then the perturbation with FGSM is computed as the sign of the model's cost function gradient. The adversarial perturbation generated with FGSM [6] is mathematically given as

$$x^{adv} = x + \epsilon sign(\nabla_x L(\theta, x, y)) \tag{1}$$

where ϵ is a scaling constant ranging between 0.0 to 1.0 is set to be very small such that the variation in x (δx) is undetectable. One can observe that in FGSM the input x is perturbed along each dimension in the direction of gradient by a perturbation magnitude of ϵ. While, a small ϵ leads to well-disguised adversarial samples, a large ϵ, is likely to introduce large perturbations.

2.1.2 *Basic Iterative Method (BIM)*

. From previous discussion it can be observed that, FGSM adds perturbation in each of the dimension, however, no optimization on perturbations are performed. In [12] Kurakin proposed an iterative version of FGSM, called as Basic iterative method (BIM). BIM is an extension of FGSM technique, where instead of applying the adversarial perturbation once with ϵ, the perturbation is applied multiple times iteratively with small ϵ. This produces a recursive noise on the input and optimized application of noise. It can be mathematically represented as follows:

$$\begin{aligned} x_0^{adv} &= x \\ x_{N+1}^{adv} &= Clip_{x,\epsilon}(x_N^{adv} + \epsilon sign(\nabla_x L(\theta, x_N^{adv}, y))) \end{aligned} \tag{2}$$

In the above mathematical expression,$Clip_{x,\epsilon}$ represents the clipping of the adversarial input magnitudes such that they are within the neighborhood of the original sample x. This technique allows more freedom for the attack compared to the FSGM method because the perturbation can be controlled and the distance of the adversarial sample from the classification boundary can be carefully fine-tuned. The experimental results presented in [12] have shown that BIM can cause higher misclassifications compared to the FGSM attack on the Imagenet samples.

2.1.3 *Momentum Iterative Method*

. The momentum method is an accelerated gradient descent technique that accumulates the velocity vector in the direction of the gradient of the loss function across multiple iterations. In this technique, the previous gradients are stored, which aids in navigating through narrow valleys of the model, and alleviate problems of

getting stuck at local minima or maxima. This momentum method also shows its effectiveness in stochastic gradient descent (SGD) to stabilize the updates. This MIM principle is applied in [4] to generate adversarial samples. MIM has shown a better transferability and shown to be effective compared to FGSM attack.

2.1.4 DeepFool Attack. DeepFool (DF) is an untargeted adversarial attack optimized for L_2 norm, introduced in [16]. DF is efficient and produces adversarial samples which are more similar to the original inputs as compared to the discussed adversarial samples generated by FGSM and BIM attacks. The principle of the Deepfool attack is to assume neural networks as completely linear with a hyper-plane separating each class from another. Based on this assumption, an optimal solution to this simplified problem is derived to construct adversarial samples. As the neural networks are non-linear in reality, the same process is repeated considering the non-linearity into the model. This process is repeated multiple times for creating the adversaries. This process is terminated when an adversarial sample is found i.e., misclassification happens. Considering the brevity and focus of the current work, we limit the details in this draft. However, the interested readers can refer to the work in [16] for exact formulation of DF.

2.2 Adversarial Defenses

So far, the different adversarial attack techniques are discussed. Here, we discuss some of the prominent existing defenses against the above discussed attacks.

2.2.1 Adversarial Training
. Adversarial training is one of the preliminary solutions for making the ML classifiers robust against the adversarial examples, proposed in [23]. The idea is to train the ML classifier with the adversarial examples so that the ML classifier can have adversarial information [6, 27] and adapt its model based on the learned adversarial data. However, one of the major drawbacks of this technique is to anticipate the type of attack and train the classifier based on those attacks and determining the criticality of the adversarial component.

2.2.2 Defensive Distillation
. Defensive distillation is another defense technique proposed in [21]. The idea is to train the classifier using the distillation training techniques and hides the gradient between softmax layer and the presoftmax layer. This makes it complex to generate adversarial examples directly on the network [3], as the knowledge is imparted from a bigger network during the training process. However, [9] shows that such a defense can be bypassed with one of the following three strategies: (1) choosing a more proper loss function; (2) calculation of gradients from pre-softmax layer rather than softmax layer; or (3) attack an easy-to-attack dummy network first and then transfer to the distilled network, similar to the distillation defense. The generation of adversaries can be simpler if the attacker knows the parameters and architecture of the defense network i.e., whitebox attack.

2.2.3 MagNet. MagNet is proposed in [14], where a two-level strategy with detector and reformer is proposed. In the detector phase(s), the system learns to differentiate between normal and adversarial examples by approximating the manifold of the normal examples. This is performed with the aid of auto-encoders. Further, in the reformer, the adversarial samples are moved close the manifold of normal samples with small perturbations. Further using the diversity metric, the MagNet can differentiate the normal and adversarial samples. MagNet is evaluated against different adversarial attacks presented previously and has shown to be robust in [14].

2.2.4 Detecting Adversaries. Another defense technique is to detect adversarial examples with the aid of statistical features [8] or separate classification networks [15]. In [15], for each adversarial technique, a DNN classifier is built to classify whether the input is a normal sample or an adversary. The detector was directly trained on both normal and adversarial examples. The detector shows good performance when the training and testing attack examples were generated from the same process and the perturbation is large enough. However, it does not generalize well across different attack parameters and attack generation processes.

subcaption

3 RRAM-BASED NEUROMORPHIC ARCHITECTURE

The Internet of Things (IoT) continues to expand and there is increasing interest in computing at the edge of the network especially in machine learning applications. Many current implementations of machine learning, and more specifically Deep Neural Networks (DNNs), have large power and computational resource requirements. The high memory bandwidth and power requirement prevent implementation in low-power real-time applications. Neuromorphic architectures have been explored for near-memory and in-memory computing to for low-power high memory bandwidth implementation of neural networks. Particularly, two-terminal RRAM devices in crossbar arrays have been extensively investigated to store weight matrix and perform in-memory computing. The RRAM-based weight matrix crossbar of neural networks can be visualized in Figure 3. The input vector gets multiplied by the weight matrix to produce the composition of the neurons at the output. A winner-take-all or softmax activation function can then be used to determine the winning neuron and thus the classification. In spite of tremendous progress in this area two-terminal RRAM devices suffers from issues such as high write current, convoluted read and write strategies, and need for a selector diode to mitigate sneak currents. Recently reported gated-RRAM devices offers to solve these issues and provide opportunities for simultaneous read and write which will be very beneficial for adjusting the weights as needed.

Gated-RRAM have been investigated as memristive synaptic devices for in-memory computing of the multiply-accumulate (MAC) behavior of neurons. Additionally, integrating multi-state gated-RRAM allows for a more dense memory crossbar since each gated-RRAM device can store multiple bits. The weight matrix crossbar of neural networks can be visualized in Figure 2. The input vector gets multiplied by the weight matrix to produce the composition of the neurons at the output. A winner-take-all or softmax activation function can then be used to determine the winning neuron and thus the classification. Similarly in a neuromorphic gated-RRAM crossbar, the product of the input voltage and gated-RRAM conductance produces a current at each synaptic branch of a neuron.

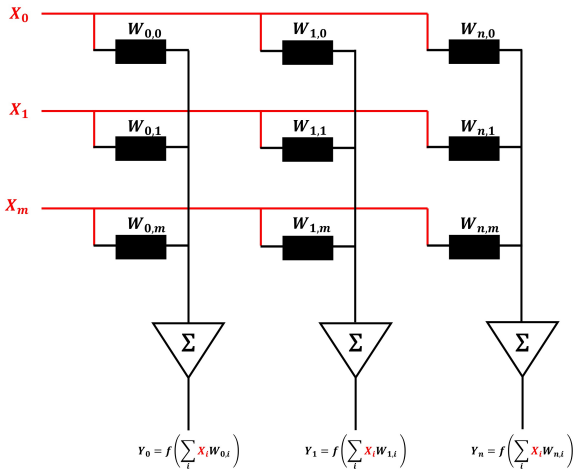


Figure 2: Feed-forward neural network crossbar performing MAC operation on input vector and weight matrix

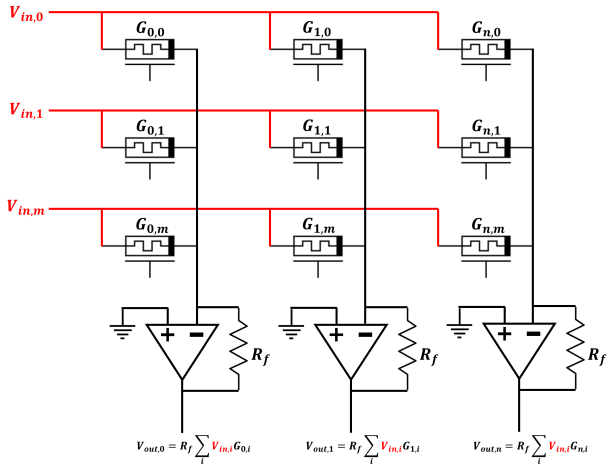


Figure 3: Gated-RRAM neuromorphic crossbar performing MAC operation on voltage-encoded input vector and conductance-encoded weight matrix

The summation amplifier at the output then computes the multiply-accumulate function of the input vector, applied as voltages and the weight vector of gated-RRAM conductances of the neuron as shown in Figure 3.

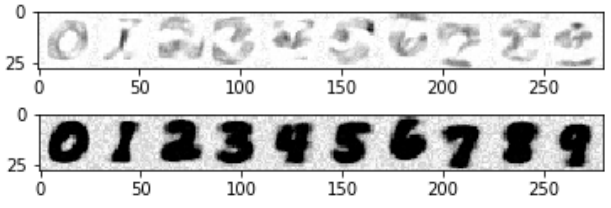


Figure 4: (a) Weight map of keras-trained weights on MNIST; (b) Weight map of STDP-based training on MNIST

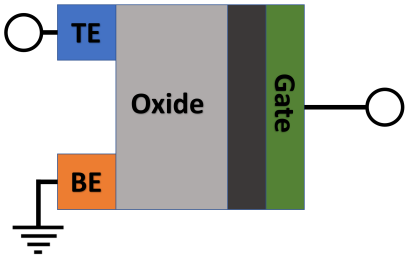


Figure 5: Gated-RRAM device model with top electrode (TE), bottom electrode (BE), gate, and channel oxide

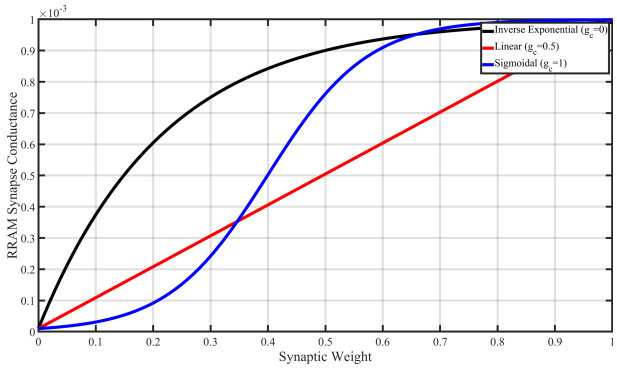


Figure 6: Conductance curves of gated-RRAM tuned through gated-synaptic model in [10]

The reconfigurability of gated-RRAM allows for synaptic weights to be programmed into the neuromorphic crossbar. The gated-RRAM device can be potentiated or depressed by applying a positive or negative bias at the gate respectively. We were able to train weights using the keras package in Python 3.9 on the MNIST dataset and then write them to the neuromorphic crossbar. However, the conductance curve of the gated-RRAM devices influences how the off-chip trained weights are mapped to the RRAM conductance space. This may effect the behavior of the neuromorphic-implemented model depending on the non-linearity of the conductance curve of the RRAM devices. Additionally, on-chip training directly on the RRAM crossbar has been explored and has demonstrated competitive training accuracy [28]. In Bailey et al. [28], a spike-timing dependent plasticity-based (STDP) algorithm is used to tune the weights of neuromorphic architecture. This training algorithm results in different weight map, thus a different trained model, than the keras-trained weights as shown in Figure 4.

Gated-resistive random-access memory (gated-RRAM) has been recently report and investigated as a gated-synaptic device for neuromorphic architectures [5]. Gated-RRAM is a memristive device consisting of a top and bottom electrode connected to an oxide channel as shown in Figure 5. A gate terminal is coupled to the channel oxide through an insulating layer allowing for a bias to be applied across the channel oxide while simultaneously applying a bias across the top and bottom electrode. This behavior allows for simultaneous reading and writing the gated-RRAM device and allows minimal programming circuitry. The gated-RRAM device

Adversarial attack	Parameter	No Attack	With Attack
FGSM	ϵ = 0.3	98.25 %	8.54
BIM	ϵ = 0.3	98.25 %	1.34
MIM	ϵ = 0.3	98.25 %	1.28
DF	MI = 50	98.25 %	1.13

Table 1: Accuracy of neural network before and after adversarial attack

can be potentiated by applying a positive bias to the gate to cause the oxygen defects to drift toward the top and bottom electrode increasing the conductance of the channel. The device can then be depressed by applying a negative bias to the gate to cause the oxygen defects to diffuse back towards the channel oxide lowering the conductance of the channel. The conductance states or curve a device follows as it is potentiated and depressed is material dependent. The conductance of these devices has been shown to follow sigmoidal, linear, or inverse exponential trends as shown in Figure 6 [10]. These conductance curves can determine the distribution of the conductance states of the gated-RRAM device and are controlled by g_c in the gated-synaptic device model [10]. Similar to the two-terminal RRAM crossbar arrays, in a neuromorphic gated-RRAM crossbar, the product of the input voltage and gated-RRAM conductance produces a current at each synaptic branch of a neuron. The summation amplifier at the output then computes the multiplyaccumulate function of the input vector, applied as voltages and the weight vector of gated-RRAM conductances of the neuron as shown in Figure 3.

4 RESULTS

In this section, we evaluate and present the performance on the MNIST digit dataset [13]. The adversarial attacks are generated using Cleverhans library [19].

Table 1 reports the performance of the employed neural network on MNIST Digits dataset. Normal Classification Accuracy: In the absence of adversarial samples, the classifier achieves an accuracy of 98.25%, loss of 0.088, precision of 0.98, and recall of 0.98. We also report the accuracy of neural network in the presence of adversarial attacks in Table 1. The number of adversarial samples are 10,000 in each case, and one can observe that in the presence of adversaries the classification accuracy falls to as low as 1.13%. With the increase in ϵ, the accuracy decreases in case of FGSM, MIM and BIM.

We present the effect of adversarial training by training neural network with adversarial data generated using different adversarial attacks and present the accuracy results in Table 2. The results show that after performing adversarial training, the accuracy of the neural network to classify data improves as reported in the Table 2. However, the accuracy can further degrade if the attack parameters (ϵ in this case) are modified.

We then loaded the keras-trained weights to the RRAM neuromorphic architecture and tested it on the MNIST dataset and the various adversarial attacks. Additionally, we also tested the performance of the on-chip STDP-trained weights on the same datasets. Table 3 report the accuracy of both the keras-trained and STDP-trained weights with RRAM devices with completely linear conductance curve and no write variability. In the absence of adversarial samples, the off-chip keras-trained network outperforms the on-chip STDP-trained weights but both training methods yield above 70% accuracy. However, with the introduction of adversarial samples, the accuracy is suddenly reduced to below 10% for keras-trained weights and below 20% for STDP-trained weights. Additionally, we tested the two networks with varying conductance curves, as shown in Table 4, and varying device write variability, as shown in Table 5. In all cases, the accuracy of the RRAM architecture was reduced to below 20% once adversarial samples were introduced.

	Attack	BIM	MIM	FGSM	DF
	Parameter	ϵ = 0.3	ϵ = 0.3	ϵ = 0.3	MI=50
Training	FGSM	79.1	78.2		77.1
	DF	53.40	54.25	41.04	
	MIM	78.2		73.7	77.1
	BIM		78.1	69.3	78.2

Table 2: Accuracy (%) of MNIST Classification under Different Adversarial Attacks on Different Adversarial Trained Networks

	Adversarial Attack				
Training	No Attack	FGSM	BIM	DF	MIM
Keras	90.44%	8.74%	8.95%	9.60%	8.91%
STDP	71.22%	13.09%	11.98%	11.69%	12.15%

Table 3: Accuracy (%) of MNIST classification of RRAM neuromorphic architecture on MNIST dataset and different adversarial attacks. The conductance curve of the gated-RRAM devices is linear and there is no write variance present.

		Adversarial Attack				
Training	RRAM g_c	No Attack	FGSM	BIM	DF	MIM
Keras	0.0	85.53%	9.54%	9.63%	10.43%	9.59%
Keras	0.5	90.44%	8.74%	8.95%	9.60%	8.91%
Keras	1.0	65.21%	4.96%	5.12%	5.77%	4.97%
STDP	0.0	72.29%	9.67%	9.08%	7.62%	9.15%
STDP	0.5	71.22%	13.09%	11.98%	11.69%	12.15%
STDP	1.0	72.00%	11.36%	10.44%	9.64%	10.50%

Table 4: Accuracy (%) of MNIST classification of RRAM neuromorphic architecture, with varying conductance distributions, on MNIST dataset and different adversarial attacks. Additionally, there is no write variance present.

		Adversarial Attack				
Training	RRAM σ^2	No Attack	FGSM	BIM	DF	MIM
Keras	0	90.44%	8.74%	8.95%	9.60%	8.91%
Keras	10^{-6}	90.42%	8.75%	8.96%	9.60%	8.92%
Keras	10^{-5}	90.38%	8.84%	9.00%	9.64%	8.99%
Keras	10^{-4}	90.41%	8.80%	9.09%	9.70%	9.01%
STDP	0	71.22%	13.09%	11.98%	11.69%	12.15%
STDP	10^{-6}	71.2%	13.08%	11.98%	11.68%	12.16%
STDP	10^{-5}	71.19%	13.10%	12.00%	11.70%	12.20%
STDP	10^{-4}	70.74%	13.26%	12.09%	11.75%	12.27%

Table 5: Accuracy (%) of MNIST classification of RRAM neuromorphic architecture, with varying device variance, on MNIST dataset and different adversarial attacks. Additionally, the RRAM devices have a linear conductance distribution.

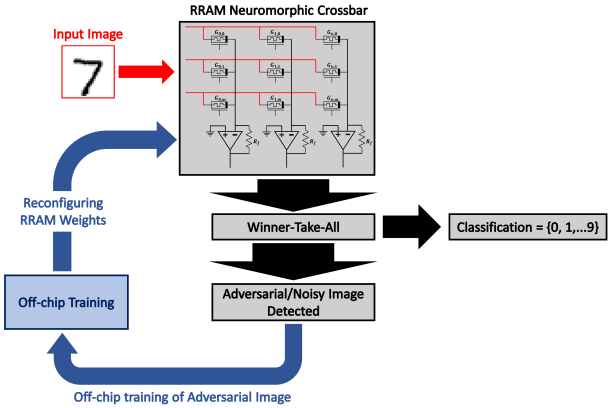


Figure 7: Integrated architecture of neuromorphic architecture with adversarial attack detection and mitigation

5 PROPOSED METHOD OF MITIGATION AND FUTURE WORK

In this work, we demonstrated that the gated-RRAM neuromorphic architecture serves as a versatile implentation of a hardware neural network. The weights can be trained off-chip using software models and programmed during run-time. Additionally, more biologically inspired algorithms can be applied to train the synaptic devices on-chip. The gated-RRAM architecture is able to tune is learned model through different training algorithms and through different device characteristic such as conductance distribution and noise from device write variance. However, we observed that this versatile model is still susceptible to an unknown adversarial attack.

We would like to investigate an integrated system as shown in Figure 7. Our results in 2 demonstrate the ability of software networks to train on adversarial attacks for future mitigation of adversarial attacks. We have also demonstrated the ability to load (write) pre-trained weights to the RRAM neuromorphic architecture model. Therefore, a software network trained on adversarial models can be implemented using the RRAM neurormorphic architectures allowing it to be deployed in low-power real-time applications while mitigating attacks from adversaries. Additionally, as shown in 7, we would like to explore methods for the chip to detect unknown adversarial attacks so that it may later train on them and identify adversarial inputs or sources. Our goal is to develop a reconfigurable architecture that is robust to adversarial attacks and with the capabilities to be integrated in low-power IoT devices or applications.

6 ACKNOWLEDGEMENT

Authors Siddharth Barve's and Rashmi Jha's efforts pertaining to gated-RRAM based neuromorphic architecture is supported by National Science Foundation under award no: CCF - 1718428.

REFERENCES

[1] Abhiroop Bhattacharjee and Priyadarshini Panda. 2020. Rethinking non-idealities in memristive crossbars for adversarial robustness in neural networks. *arXiv preprint arXiv:2008.11298* (2020).
[2] Battista Biggio, Blaine Nelson, and Pavel Laskov. 2012. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389* (2012).
[3] Nicholas Carlini and David A. Wagner. 2017. Towards Evaluating the Robustness of Neural Networks. *2017 IEEE Symposium on Security and Privacy (SP)* (2017), 39–57.
[4] Y. Dong, Fangzhou Liao, Tianyu Pang, H. Su, J. Zhu, Xiaolin Hu, and J. Li. 2018. Boosting Adversarial Attacks with Momentum. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), 9185–9193.
[5] T. Bailey E. Herrmann, A. Rush and R. Jha. April 2018. Gate Controlled Three-Terminal Metal Oxide Memristor. *IEEE Electron Device Letters* vol. 39, no. 4, pp. 500-503 (April 2018).
[6] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).
[7] Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. 2019. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics* (11 2019).
[8] Kathrin Grosse, P. Manoharan, Nicolas Papernot, M. Backes, and P. McDaniel. 2017. On the (Statistical) Detection of Adversarial Examples. *ArXiv* abs/1702.06280 (2017).
[9] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).
[10] A. Jones and R. Jha. [n.d.]. A Compact Gated-Synapse Model for Neuromorphic Circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 10.1109/TCAD.2020.3028534 ([n. d.]).
[11] Edward Kim, Jessica Yarnall, Priya Shah, and Garrett T Kenyon. 2019. A neuromorphic sparse coding defense to adversarial images. In *Proceedings of the International Conference on Neuromorphic Systems*. 1–8.
[12] Alexey Kurakin, I. Goodfellow, and S. Bengio. 2017. Adversarial examples in the physical world. *ArXiv* abs/1607.02533 (2017).
[13] Yann LeCun and Corinna Cortes. 2010. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/. (2010). http://yann.lecun.com/exdb/mnist/
[14] Dongyu Meng and Hao Chen. 2017. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*. 135–147.
[15] J. H. Metzen, Tim Genewein, Volker Fischer, and B. Bischoff. 2017. On Detecting Adversarial Perturbations. *ArXiv* abs/1702.04267 (2017).
[16] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and P. Frossard. 2016. DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), 2574–2582.
[17] Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C Lupu, and Fabio Roli. 2017. Towards poisoning of deep learning algorithms with back-gradient optimization. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. 27–38.
[18] Blaine Nelson, Marco Barreno, F. J. Chi, A. Joseph, Benjamin I. P. Rubinstein, Udam Saini, Charles Sutton, J. Tygar, and Kai Xia. 2008. Exploiting Machine Learning to Subvert Your Spam Filter. In *LEET*.
[19] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, I. Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, T. Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, W. Gierke, Y. Dong, David Berthelot, P. Hendricks, Jonas Rauber, Rujun Long, and P. McDaniel. 2016. Technical Report on the CleverHans v2.1.0 Adversarial Examples Library. *arXiv: Learning* (2016).
[20] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. 2016. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*. IEEE, 372–387.
[21] Nicolas Papernot, P. McDaniel, Xi Wu, S. Jha, and A. Swami. 2016. Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks. *2016 IEEE Symposium on Security and Privacy (SP)* (2016), 582–597.
[22] Benjamin IP Rubinstein, Blaine Nelson, Ling Huang, Anthony D Joseph, Shing-hon Lau, Satish Rao, Nina Taft, and J Doug Tygar. 2009. Antidote: understanding and defending against poisoning of anomaly detectors. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*. 1–14.
[23] Uri Shaham, Yutaro Yamada, and Sahand Negahban. 2015. Understanding adversarial training: Increasing local stability of neural nets through robust optimization. *arXiv preprint arXiv:1511.05432* (2015).
[24] S. Shukla and et al. 2019. Stealthy malware detection using rnn-based automated localized feature extraction and classifier. In *ICTAI*. IEEE.
[25] S. Shukla and et al. 2019. Work-in-Progress: MicroArchitectural Events and Image Processing-based Hybrid Approach for Robust Malware Detection. In *CASES*.
[26] Jack Stilgoe. 2017. Machine learning, social learning and the governance of self-driving cars. *Social Studies of Science* (11 2017).
[27] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).
[28] S. Barve J. Wells T. J. Bailey, A. J. Ford and R. Jha. [n.d.]. Development of a Short-Term to Long-Term Supervised Spiking Neural Network Processor. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* doi: 10.1109/TVLSI.2020.3013810 ([n. d.]).