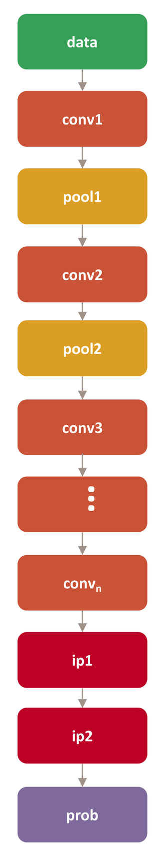
SoWaF: Shuffling of Weights and Feature Maps: A Novel Hardware Intrinsic Attack (HIA) on Convolutional Neural Network (CNN)

Tennessee TECH

Tolulope A. Odetola and Syed Rafay Hasan

I. INTRODUCTION

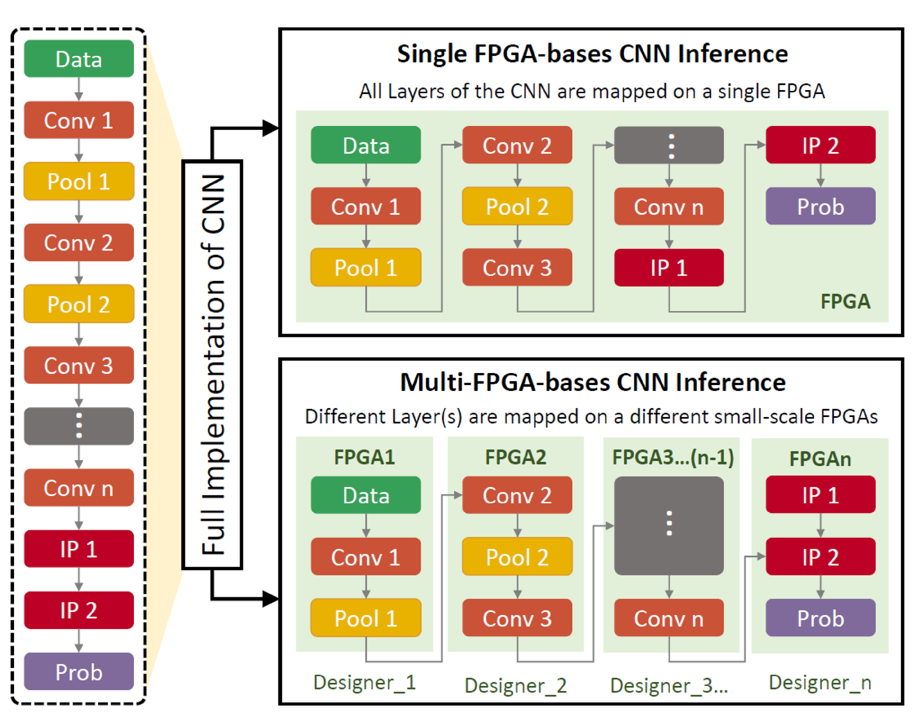
- FPGA hardware accelerators offer good performance, high energy efficiency, fast prototyping, and capability of reconfiguration.
- To achieve short time-to-market, the mapping of pre-trained CNN on hardware accelerators is often outsourced to untrusted third parties.
- Due to their untrusted nature hardware intrinsic security can be compromised via malicious hardware insertions, which are very difficult to detect, especially if the IP is provided as a bitstream file.



II. Problem Formulation

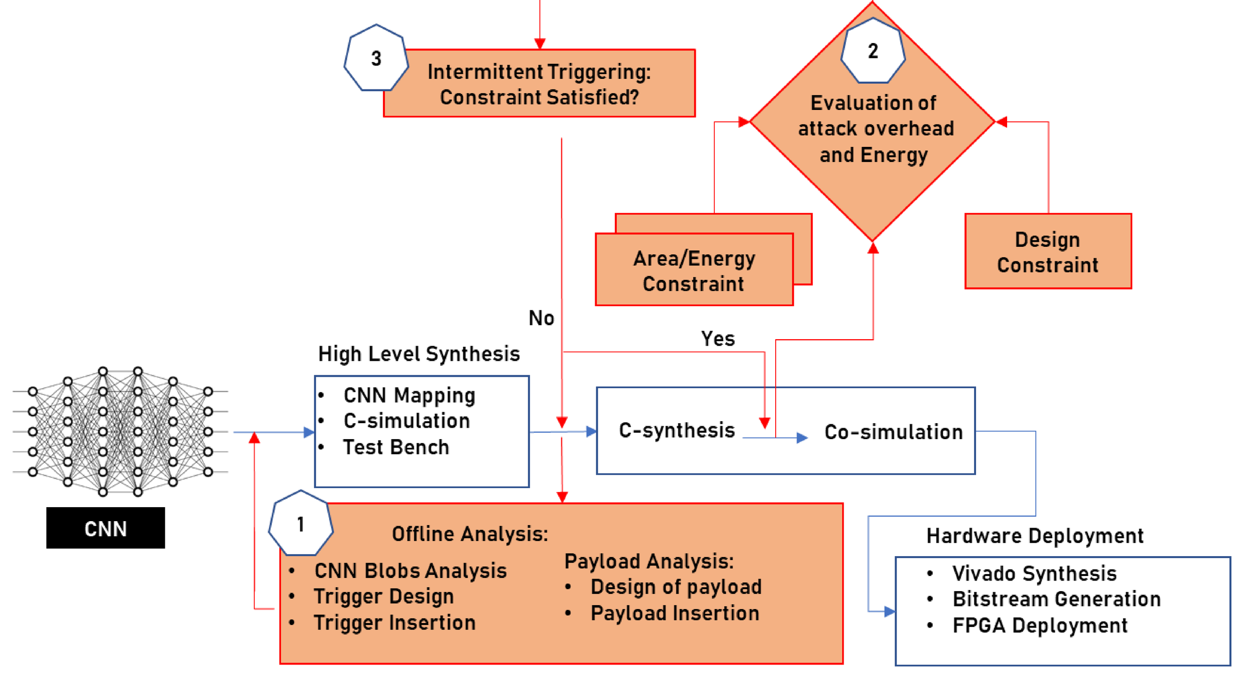
Different techniques of inserting hardware attacks into CNNs have been explored. These techniques assume:

- These attacks require a manipulation of the CNN parameters.
- The attacker has full knowledge of the CNN architecture.
- The trigger is dependent on the input image
- Their payload require extra computation
- The attack is designed for a single FPGA based inference.



- In a situation where the full CNN architecture is not accessible to any one designer as seen in Multi-FPGA CNN inference. The approaches in literature may not be applicable.
- In this work we propose a framework of attack called SoWaF (Shuffling of Weights and Feature Maps) that leads to misclassification applicable to single and multi-FPGA CNN inference.
- This approach does not require full access to the CNN architecture.

III. SoWaF Approach

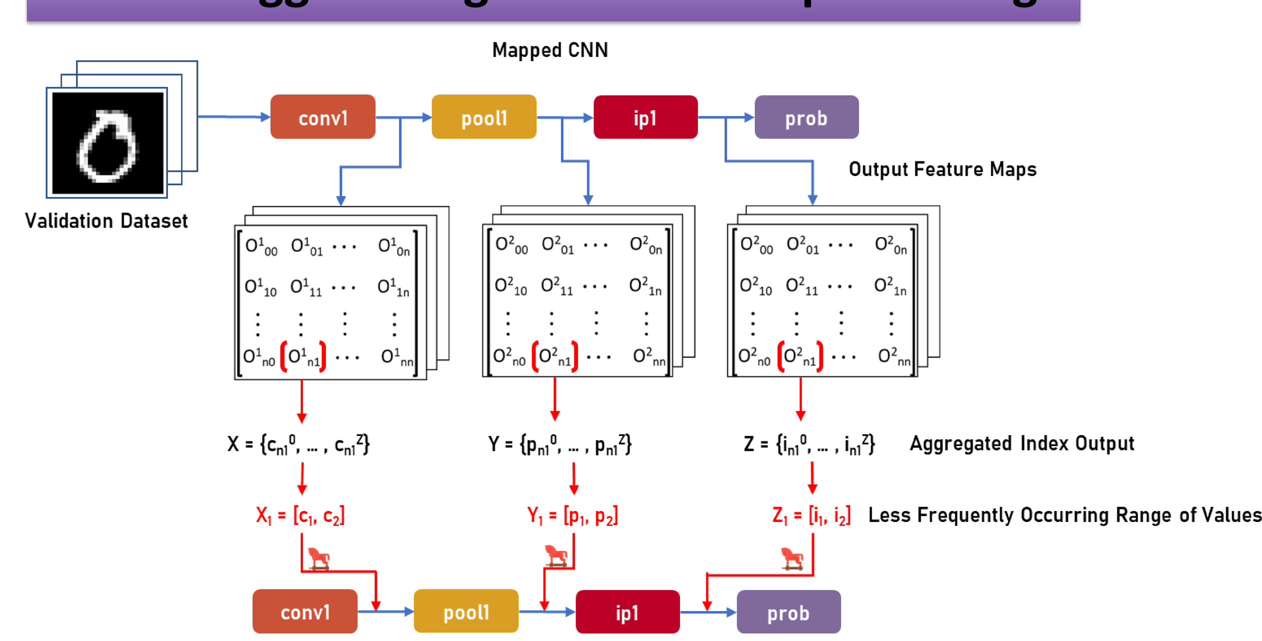


Overview of SoWaF (trigger and payload) methodology flow is shown above.

- **Section 1** of the methodology flow involves the offline analysis of the output feature maps to design a stealthy trigger.
- **Section 2** shows the comparison of the additional hardware overhead incurred by the embedded attack circuitry with the design constraints.
- **Section 3** shows the evaluation of the stealthiness and effectiveness of the attack.

II. Methodology

SoWaF Trigger Design: Offline Pre-processing

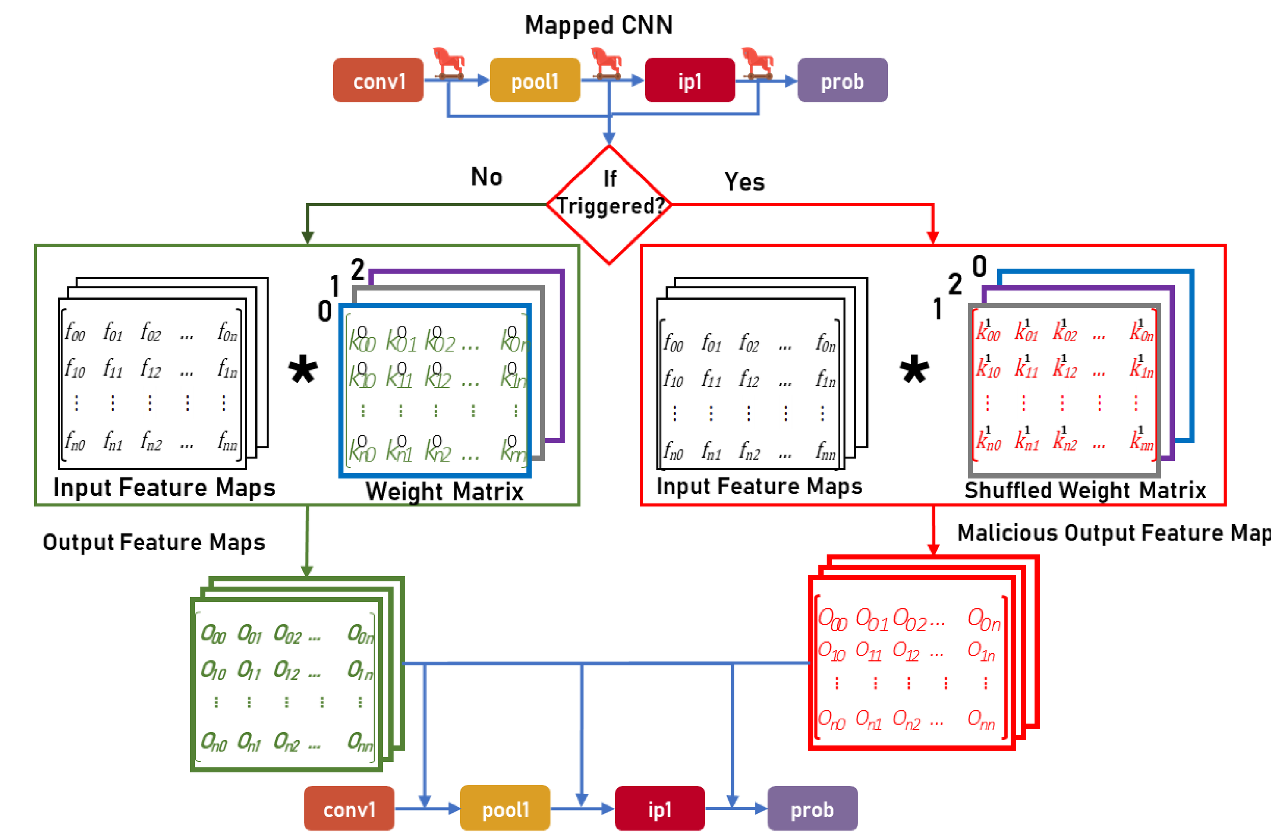


The attacker collects the output feature maps to setup a trigger.

- As shown in the diagram above, during the functional verification stage, a validation dataset can be used by the attacker to access the respective CNN layer's output feature maps for all the dataset.
- By choosing an index randomly of one of the channels of the output feature map of any chosen CNN layer as shown above.
- The attacker can monitors the values (X or Y or Z) of the randomly selected index to obtain a generalized range of values (RoV)
- The selected RoV for a given CNN layer serve as the trigger for the attack.

II. Methodology Cont'd

SoWaF Payload Design: Runtime Operation



- Upon triggering, for convolution and fully connected layers, the payload shuffles the channels of the weight matrix with another one as illustrated on the right hand side of the decision block above.
- CNN layers other than convolution and fully connected layers (such as Pooling layer, etc.) do not have weight matrices and channels, the storage of the output feature maps are shuffled
- This leads to miscalculation in the layer hence leading to the layer output and consequently misclassification

IV. Result

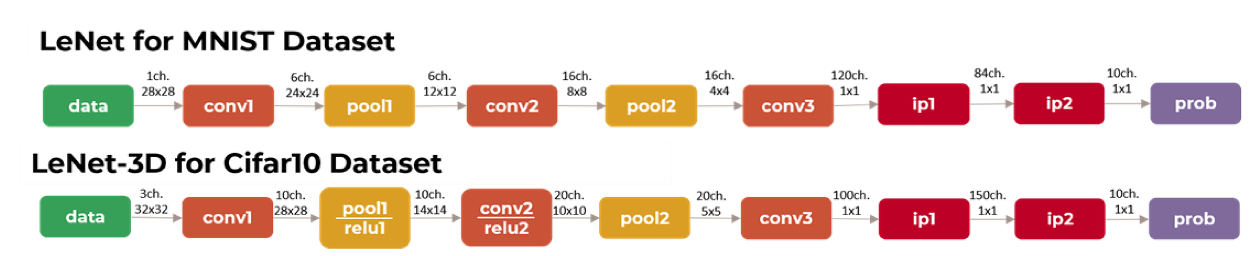


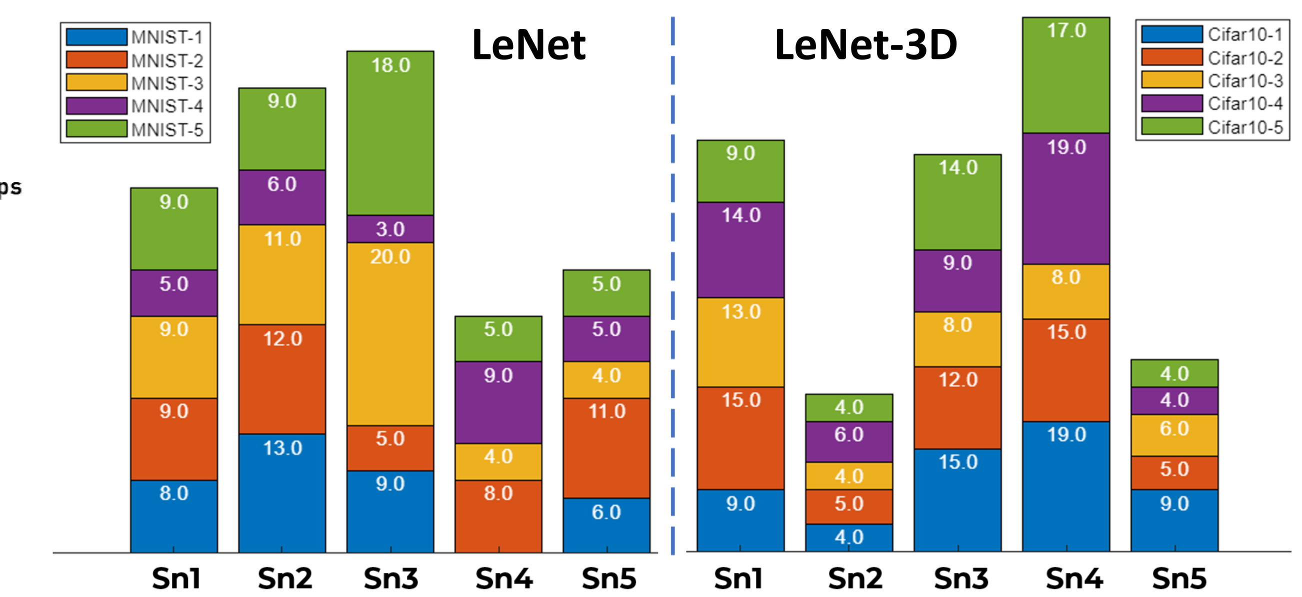
TABLE I: Resource overhead comparison between attacks on different layers of LeNet and LeNet-3D compared to their respective originals

Network	Attack Scenario (Sn): Layer	Chs	BRAM	% diff	DSPs	% diff	LUTs (x1000)	% diff	FFs (x1000)	% diff	Latency (x1000) clock-cycles	% diff
LeNet	Original	-	42	-	33	0	118.5	-	58.3	-	680.4	-
	Sn1: *conv*1 attack	6	42	0	33	0	119.2	+0.61	59.2	+1.5	680.51	+0.003
	Sn2: *pool*1 attack	6	42	0	33	0	118.9	+0.34	58.8	+0.76	680.51	+0.003
	Sn3: *conv*2 attack	16	53	+26	33	0	121.3	+2.36	58.8	+0.81	680.58	+0.013
	Sn4: *pool*2 attack	16	42	0	33	0	119.2	+0.34	59.3	+0.76	680.51	+0.003
	Sn5: *conv*3 attack	120	162	+285	33	0	780.7	-34	34.5	-41	680.74	+0.038
LeNet-3D for Cifar10	Original	-	59	-	37	-	49.0	-	39.7	-	1685.71	-
	Sn1: *conv*1 attack	5	59	0	37	0	49.9	+1.81	40.5	+1.8	1685.73	+0.001
	Sn2: *pool*1 attack	5	59	0	37	0	49.6	+1.16	40.4	+1.76	1685.72	+0.001
	Sn3: $conv2 attack	20	79	+34	37	0	48.6	-0.78	39.0	-1.9	1685.72	+0.001
	Sn4: *pool*2 attack	20	59	0	37	0	50.0	+1.93	41.0	+3.2	1695.99	+0.61
	Sn5: *conv*3 attack	100	159	+169	37	0	20.1	-59	10.0	-74.6	1685.72	+0.001

- The attack is implemented on Lenet trained on MNIST dataset and LeNet-3D for Cifar10 datasets as shown above.
- To evaluate the SoWaF attack, we propose 5 different scenarios, where each layer (from conv1 to conv3)is infected with the attack.
- From the Table above, we see that DSP and BRAM usage remains the same except for Sn3 and Sn5, where BRAM is increased (5th column in Table I).
- For LUTs and FFs in all the scenarios, other than Sn5, (i.e. Sn1-Sn4) have a very modest increment in usage (up to 2.36%).

IV. Result Cont'd

To demonstrate the randomness of SoWaF, various random datasets are examined. In the diagram below, from Sn1, when five sets (200 images each) of data is provided to LeNet and LeNet-3D, the number of trigger occurrences vary randomly between 5 to 9. Same is true for other attack scenarios- making the SoWaF attack random and stealthy



VI. CONCLUSION

The SoWaF attack achieves misclassification when triggered by shuffling the weight matrices of convolution layers to propagate wrong feature maps. This attack is carried out without changes in the model parameters. Our results for two CNN architectures show that in all the attack scenarios, additional latency is negligible (<0.61%), increment in DSP, LUT, FF is also less than 2.36%. Three of the five investigated scenarios show very minimal changes in BRAM.

VI. ACKNOWLEDGMENT

This research is partially funding provided by Tennessee Tech University College of Engineering for achieving Carnegie classification.

VII. REFERENCES

[1] K. Abdelouahab, M. Pelcat, J. Serot, and F. Berry, "Accelerating cnn inference on fpgas: A survey,"arXiv preprint arXiv:1806.01683, 2018.
[2] K. Guo, S. Zeng, J. Yu, Y. Wang, and H. Yang, "[dl] a survey of fpga-based neural network inference accelerators, "ACM Transactions on Reconfigurable Technology and Systems (TRETS), vol. 12, no. 1, pp.1–26, 2019.
[3] T. A. Odetola, K. M. Groves, and S. R. Hasan, "2l-3w: 2-level 3-way hardware-software co-verification for the mapping of deep learning architecture (dla) onto fpga boards," arXiv preprint arXiv:1911.05944,2019.
[4] M. T. Hailesellasie and S. R. Hasan, "Mulnet: A flexible cnn processor with higher resource utilization efficiency for constrained devices,"IEEE Access, vol. 7, pp. 47 509–47 524, 2019.
[5] M. Hailesellasie, S. R. Hasan, and O. A. Mohamed, "Mulmapper: towards an automated fpga-based cnn processor generator based ona dynamic design space exploration," in2019 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, 2019, pp. 1–5.
[6] X. Wei, C. H. Yu, P. Zhang, Y. Chen, Y. Wang, H. Hu, Y. Liang,and J. Cong, "Automated systolic array architecture synthesis for high throughput cnn inference on fpgas," in Proceedings of the 54th Annual Design Automation Conference 2017, 2017, pp. 1–6.
[7] J. H. Kim, B. Grady, R. Lian, J. Brothers, and J. H. Anderson,"Fpga-based cnn inference accelerator synthesized from multi-threaded csoftware," in2017 30th IEEE International System-on-Chip Conference(SOCC). IEEE, 2017, pp. 268–273.