Genetic Improvement for DNN Security

Hunter Baxter
Vanderbilt University
Nashville, USA
hunter.c.baxter@vanderbilt.edu

Yu Huang
Vanderbilt University
Nashville, USA
yu.huang@vanderbilt.edu

Kevin Leach
Vanderbilt University
Nashville, USA
kevin.leach@vanderbilt.edu

ABSTRACT

Genetic improvement (GI) in Deep Neural Networks (DNNs) has traditionally enhanced neural architecture and trained DNN parameters. Recently, GI has supported large language models by optimizing DNN operator scheduling on accelerator clusters. However, with the rise of adversarial AI, particularly model extraction attacks, there is an unexplored potential for GI in fortifying Machine Learning as a Service (MLaaS) models. We suggest a novel application of GI — not to improve model performance, but to diversify operator parallelism for the purpose of a moving target defense against model extraction attacks. We discuss an application of GI to create a DNN model defense strategy that uses probabilistic isolation, offering unique benefits not present in current DNN defense systems.

KEYWORDS

Computer Security, Genetic Improvement

ACM Reference Format:
Hunter Baxter, Yu Huang, and Kevin Leach. 2024. Genetic Improvement for DNN Security. In *2024 ACM/IEEE International Workshop on Genetic Improvement (GI '24), April 16, 2024, Lisbon, Portugal.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3643692.3648261

1 INTRODUCTION

For complex tasks like text generation and image recognition, Deep Neural Networks (DNNs) have become ubiquitous [1, 2]. Machine Learning as a Service (MLaaS) has emerged as a popular business model, whose value is broadly determined by the model architecture and dataset. Therefore, a significant risk to MLaaS is the unauthorized disclosure of details from its training data or the DNN design, including elements such as operators, weights, dimensions, and configuration. The repercussions of data breaches are tangible: they can lead to breaches of privacy laws, ethical issues in handling personal data, and the loss of a competitive edge in datasets. Similarly, the ramifications of model theft can be drastic. A stolen model allows an entity to replicate the model with minimal research and development investment. Moreover, access to the model accelerates the discovery of other malicious applications like data poisoning, model inversion, membership inference, and the creation of adversarial examples. Attacks through model reversal or membership detection can reveal details about user training sets, infringing on user privacy. Attacks using adversarial instances might compromise the safety of autonomous vehicles, posing a significant danger to human life. Due to these hazards, model extraction attacks have been identified as a major concern by industry leaders, second only to data poisoning, which is further facilitated by access to precise model details [5]. From the perspective of national defense, the theft of a DNN used for military applications might equate to the loss of advanced weaponry technology.

Strong isolation through trusted execution environments (TEE) and data oblivious algorithms are among the most popular defense strategies for model extraction attacks from a systems perspective [6, 9]. The benefit of strong isolation is that it provides guarantees on the isolation of memory from traditional vulnerabilities, with the exception of side channel attacks. However, strong isolation necessitates considerable performance overhead and restricts the maximum feasible size of the model to fit in a TEE. Data oblivious algorithms typically share the same downsides as strong isolation, as they also use TEE, but add uniform treatment of data to avoid timing side channels. Strong isolation and data oblivious algorithms both have scalability limits, which is a challenge to secure all modern large language models. Additionally, vulnerabilities to side-channel attacks are relevant, despite their cost, considering the resources an attacker would invest to extract a valuable artifact such as a model.

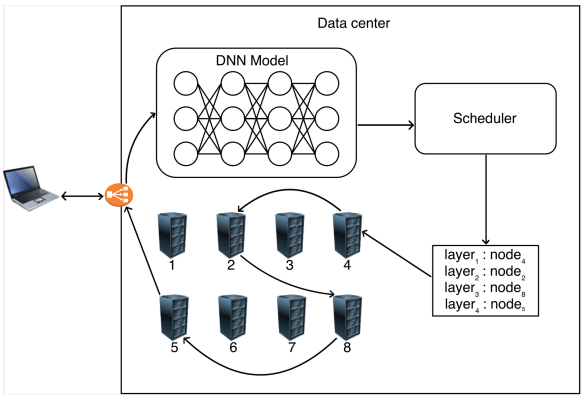

Figure 1: Proposed workflow for a moving target defense scheme. Given a cluster of worker servers, and an initial inter-operator parallelism schedule, we propose using a genetic algorithm to find similar schedules in terms of performance, but different target schedules in regards to where specific layers are in memory. Inference on a neural network is then forwarded through the scheduled nodes. By introducing uncertainty in where parameters are in memory, we posit that an attacker needs more resources to extract the model.

GI '24, April 16, 2024, Lisbon, Portugal

ACM ISBN 979-8-4007-0573-1/24/04
https://doi.org/10.1145/3643692.3648261

We propose modifying state-of-the-art automated inter-operator parallelism techniques [4, 11] to provide a moving target defense for DNN to defend against model extraction attacks. A moving target defense scheme would provide probabilistic pseudo-isolation to decrease the time value of information necessary to launch a model extraction attack. By trading the strong privacy guarantees of trusted execution environments or data-oblivious computation for probabilistic pseudo-isolation, we can secure larger models with fewer performance sacrifices, and without any hardware modification. In order to move the target effectively, we propose using genetic algorithms so that we create schedules which maintain optimal performance characteristics, but also provide isolation through a multi-objective fitness function.

2 PROBABILISTIC ISOLATION

Assumptions: When a large DNN model is served for inference, it is located on either a single server or potentially multiple servers if a parallelism scheme is used. In this setup, the model weights' location remains fixed for the duration of the service. If an attacker learns any information about the model weights' location in memory, they can repeat the successful approach until the entire model is extracted. This has been shown to work through micro-architectural vulnerabilities such as speculative execution [10], DRAM disturbance [7], and GPU timing side channels [3]. It is likely that more vulnerabilities will surface in the future. However, these attacks take considerable time, require research on specific hardware, and often necessitate launching a prior attack to learn the architecture of the model before extracting the weights.

Proposed Workflow: If the specific location of the weights in memory were moved across different nodes in a cluster, an attacker would need to duplicate their work. Suppose we have a pool of m worker nodes, with each responsible for one forward pass of a section of a neural network consisting of k layers. It can be shown that in this formulation, there are $\sum_{i=1}^{m} \binom{m}{i} i! \binom{k-1}{i-1}$ possible arrangements of node to layer set mappings, forming a schedule. In an average-case scenario, this implies a linear increase in the amount of time required for an attacker to eventually retrieve every single layer of a network if they have only compromised a single node. A linear increase in the attacker's workload is beneficial because, in a data center, thousands of nodes are used [8], leading to thousands of possible schedules. This would result in an effort likely impossible for an attacker to overcome within a timeframe before an intrusion detection system discovers it. This forms a moving target defense, where it is likely that information is isolated, called probabilistic isolation, most popularly from address space layout randomization.

Moving Target Defense: We propose using a genetic algorithm to solve a multi-objective optimization problem, balancing the similarity between the current solution and a subsequent solution, and a function that measures the respective solution's performance. By modeling the placement of DNN layers onto nodes as a bit matrix, as shown in 2, and obtaining initial measurements of network speeds for tensor transfers among servers, we can quickly evaluate the fitness of a proposed solution. By solving this multi-objective fitness function, a new solution that is similarly performant but moves the target, would provide security benefits with little cost. Moving the target in this manner could be triggered by existing intrusion detection mechanisms, a fixed time duration, or manual intervention from a human operator.

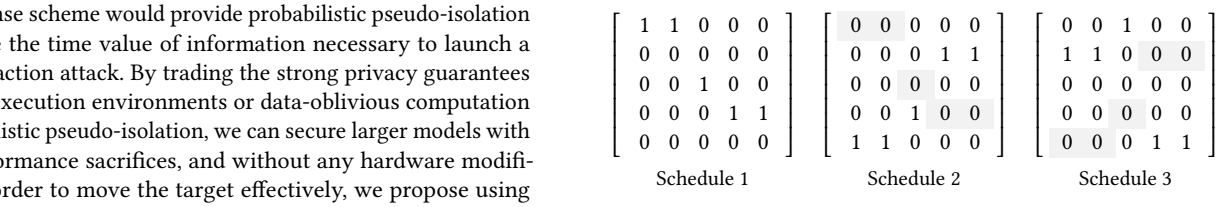

Figure 2: Example schedule matrix for 5 nodes and 4 layers, with rows and columns corresponding respectively. A 1 in position i, j of the matrix indicates that layer j in the DNN is scheduled on node i. A gray box represents the previous schedule location.

3 CONCLUSION

Adversarial AI, specifically model extraction attacks, is a growing concern. To account for the growing need to secure large-scale DNN models, a low-overhead and scalable defense solution is necessary. We suggest collaboration with the GI and security community to investigate approaches to use probabilistic isolation and genetic algorithms to provide security without the sacrifices associated with strong isolation and data-oblivious algorithms.

REFERENCES

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[3] Xing Hu, Ling Liang, Shuangchen Li, Lei Deng, Pengfei Zuo, Yu Ji, Xinfeng Xie, Yufei Ding, Chang Liu, Timothy Sherwood, et al. 2020. Deepsniffer: A dnn model extraction framework based on learning architectural hints. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*. 385–399.

[4] Sheng-Chun Kao and Tushar Krishna. 2020. Gamma: Automating the hw mapping of dnn models on accelerators via genetic algorithm. In *Proceedings of the 39th International Conference on Computer-Aided Design*. 1–9.

[5] Ram Shankar Siva Kumar, Magnus Nyström, John Lambert, Andrew Marshall, Mario Goertzel, Andi Comissoneru, Matt Swann, and Sharon Xia. 2020. Adversarial machine learning-industry perspectives. In *2020 IEEE Security and Privacy Workshops (SPW)*. IEEE, 69–75.

[6] Rishabh Poddar, Ganesh Ananthanarayanan, Srinath Setty, Stavros Volos, and Raluca Ada Popa. 2020. Visor: Privacy-preserving video analytics as a cloud service. In *Proceedings of the 29th USENIX Conference on Security Symposium*. 1039–1056.

[7] Adnan Siraj Rakin, Md Hafizul Islam Chowdhuryy, Fan Yao, and Deliang Fan. 2022. Deepsteal: Advanced model extractions leveraging efficient weight stealing in memories. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1157–1174.

[8] Arjun Roy, Hongyi Zeng, Jasmeet Bagga, George Porter, and Alex C Snoeren. 2015. Inside the social network's (datacenter) network. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*. 123–137.

[9] Stavros Volos, Kapil Vaswani, and Rodrigo Bruno. 2018. Graviton: Trusted Execution Environments on GPUs.. In *OSDI*. 681–696.

[10] Mengjia Yan, Christopher Fletcher, and Josep Torrellas. 2020. Cache telepathy: Leveraging shared resource attacks to learn DNN architectures. In *USENIX Security Symposium*.

[11] Lianmin Zheng, Zhuohan Li, Hao Zhang, Yonghao Zhuang, Zhifeng Chen, Yanping Huang, Yida Wang, Yuanzhong Xu, Danyang Zhuo, Eric P Xing, et al. 2022. Alpa: Automating inter-and {Intra-Operator} parallelism for distributed deep learning. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*. 559–578.