Extracting Information from Machine Learning Models

Iheb Ghanmi, Lars Wüstrich*
*Chair of Network Architectures and Services
School of Computation, Information and Technology, Technical University of Munich, Germany
Email: iheb.ghanmi@tum.de, wuestrich@net.in.tum.de

Abstract—**In the era of data-driven decision-making, machine learning models, especially neural networks, demonstrate their capabilities across various domains. However, as the deployment of these models increases, the vulnerability of these models to attacks has become a significant concern. This paper illustrates how attackers can extract sensitive information from machine learning models, potentially compromising the confidentiality of training data. We introduce the fundamentals of neural networks, emphasize the architecture of Feedforward Neural Networks, and explain how weights and biases intrinsically store knowledge. We present two attacks designed to extract information about the training data from a black-box neural network.**

Index Terms—**neural networks, adversarial attacks, information extraction, privacy**

1. Introduction

Machine learning (ML) currently undergoes a transformative evolution. Transitioning from an academic curiosity, it now serves as a pivotal tool in numerous real-world applications [1]. Present capabilities include detecting patterns in images [2], decoding nuances of human language [3], and recognizing intricate auditory cues [4]. Typically, these models are encapsulated and operate as "black-boxes". In this configuration, users access the input-output relationships but remain uninformed about internal operations [5]. This design choice simplifies user interactions and, crucially, safeguards sensitive data included in the training datasets [6].

However, the perceived simplicity and security hide inherent challenges. Our review reveals that over-reliance on the obscured nature of black-box models for security is problematic. Despite their strengths, neural networks (NNs) are susceptible to attacks that aim to reveal information, particularly regarding the data they were trained on [7]. Most crucially, and as our primary contribution, we examine various techniques, shedding light on the **inherent weaknesses** of these networks, challenging their perceived *invulnerability*, and underscoring the **urgent** need for better defenses [8].

The remainder of this paper is structured as follows: Section 2 introduces the basics of NNs, Section 3 defines our threat model, and Section 4 delves into methods for information extraction. In Section 5, we explore specific applications of the attacks within the networking domain, especially in Intrusion Detection Systems (IDS). We conclude with a discussion on future directions.

2. NN Basics

This section introduces concepts related to NNs that are essential for understanding subsequent discussions for attacks and defenses. An NN is a computational model inspired by biological NNs in the human brain. The primary function of an NN is to receive input, process it, and provide an output.

2.1. NN Architecture

The architecture of an NN defines its fundamental structure, detailing how individual components, such as neurons, are interconnected. This structure plays a pivotal role in determining the network's computational capabilities and its ability to learn from data.

Neurons: Neurons are fundamental units in an NN. A neuron receives multiple inputs, processes them, and generates a single output. This processing involves a **weighted** sum of the inputs, an addition of a **bias**, and the application of an activation function [9].

Activation Functions: These are mathematical functions that, given an input, determine the output of a neuron. Common activation functions include the sigmoid, tanh [10], and ReLU (Rectified Linear Unit) [11].

Layers: Typically, we organize NNs in layers [9]. The three main types of layers are:

Input Layer: This is where the network receives input from the dataset. Each neuron in this layer corresponds to one feature in the dataset.

Hidden Layer: These are layers between the input and output layers and are each composed of multiple neurons. An NN can have any number of hidden layers, and this is what makes a network "deep" in deep learning [12].

Output Layer: This layer produces the final prediction or classification of the network.

Feedforward NNs (FNNs): FNNs represent the most straightforward artificial NN architecture type [10]. In FNNs, the data flows in one direction, from the input layer, through the hidden layers, and to the output layer. There are no cycles or loops in the network. Figure 1 provides an overview of such a network.

2.2. Knowledge in NNs

NNs store knowledge as weights and biases. Weights determine the connection strength between two neurons. Biases, similar to intercepts in linear equations, allow neuron output adjustments. During training, the network

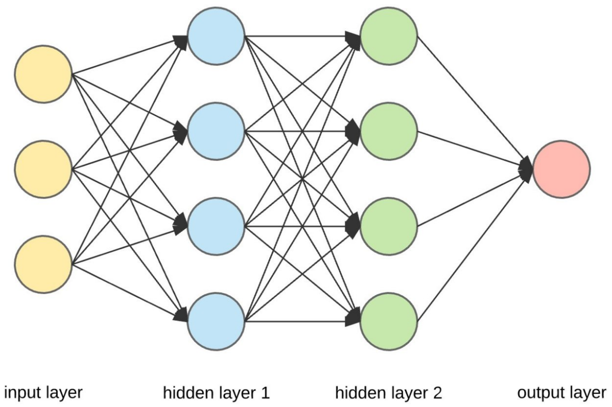

Figure 1: Overview of a simple example structure of an FNN of one input layer, two different hidden layers and one output layer consisting of one neuron. Model was adapted from [13]

modifies its weights and biases to reduce prediction discrepancies from actual outcomes, typically using backpropagation and optimization techniques like gradient descent [9]. A common challenge with NNs is their black-box nature. Although they produce relatively accurate predictions, explaining the exact reasoning behind specific decisions remains difficult [14]. The knowledge in the network is distributed across the NN's weights and biases, and it is not always clear how individual weights contribute to the final decision.

3. Threat Model

This section defines our threat model which consists of the properties of our target model as well as the attacker's capabilities.

Target Model: The target model refers to any trained machine learning model, particularly those that have been trained on sensitive datasets, such as medical records or personal information. These models range from deep NNs [15] to classification models trained by popular "machine learning as a service" providers [16].

Attacker Capabilities:
* **Model Access:** The attacker can access the model, meaning they can send input data to the model and receive the corresponding outputs. This does not imply that the attacker has access to the original training data or any metadata associated with it.
* **Input Data:** The attacker can provide any input data to the model and observe its predictions or classifications. This allows the adversary to infer information about the model's training data or its internal workings.
* **Model Queries:** The attacker can make unlimited queries to the model. This means that the attacker can send an unlimited number of input data points to the model and observe the corresponding outputs.

Attacker Limitations:
* **Black-box Assumption:** Although the attacker interacts with the model, they do not have direct access to the model's internal parameters, weights, or architecture. This means that the attacker cannot directly observe or manipulate the inner workings of the model.
* **No Training Data Access:** The attacker does not have access to the original training data or any associated metadata. This is particularly relevant in scenarios where the training data is sensitive or confidential.

4. Information Extraction Methods

This section focuses on attacks that aim to extract information about the training data of the NNs. The goal is to analyze and compare the methods that were proposed in the literature. The discussion starts with introducing the different methods and then comparing them in terms of their efficiency and accuracy. The section also discusses the different assumptions that were made by the authors of the different methods.

In this paper, the focus is on the following methods: Membership Inference Attacks in Section 4.1 and Knowledge Extraction Attacks with No Observable Data in Section 4.2. These represent two possible methods to extract information about the training data.

4.1. Membership Inference Attacks (MIAs)

Membership Inference is not a singular adversarial attack but rather represents a broader category of such attacks. In these attacks, the objective is to determine whether a specific data point belongs to the training dataset by interacting with a model via a black-box method. This method circumvents the need to rely on explicit statistics or specific details about the target model's architecture. Through this technique, the attacker learns actual information about whether a specific data point is part of the model's original training dataset.

In this context, the discussion revolves around the attack methodology introduced by Shokri et al. [16]. Here, the authors trained an **attack model** to distinguish the target model's responses based on whether the input data is part of its original training dataset (see Figure 2).

Membership inference attacks, as explored by Shokri et al. [16], utilize a technique termed shadow training. In this approach, multiple **shadow models** are constructed to mimic the behavior of the target model. These shadow models are trained on datasets that closely resemble the distribution of the target model's training data (Figure 3). A salient feature of these shadow models is the attacker's awareness of their training datasets, ensuring a clear understanding of data record membership.

This knowledge facilitates the training of the attack model using the input-output pairs from these shadow models. While various strategies can be employed to generate this data, it's paramount that the data distribution aligns closely with that of the target model's training set. By contrasting the behavior of the shadow models on their known training data with their behavior on unfamiliar inputs, the attack model can discern nuanced differences in the target model's responses.

The study highlights that, even without prior assumptions about the distribution of the target model's training data, and using fully synthetic data for shadow models, membership inference accuracy can reach up to 90% [16]. Furthermore, the research underscores the potential risks to datasets, such as those from health care, when used to

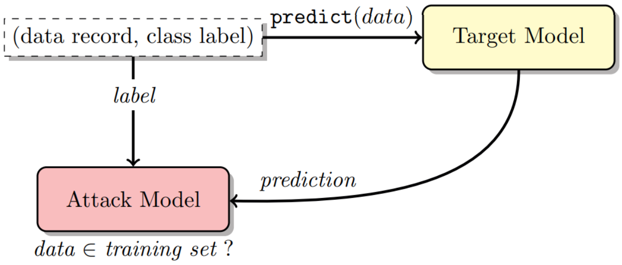

Figure 2: Overview of the Membership Inference Attack adapted from [16]. The *attack model* receives, along with the class label of the input, the prediction output of the original *target model*. A classification is then made to ascertain whether the input data was part of the original training dataset.

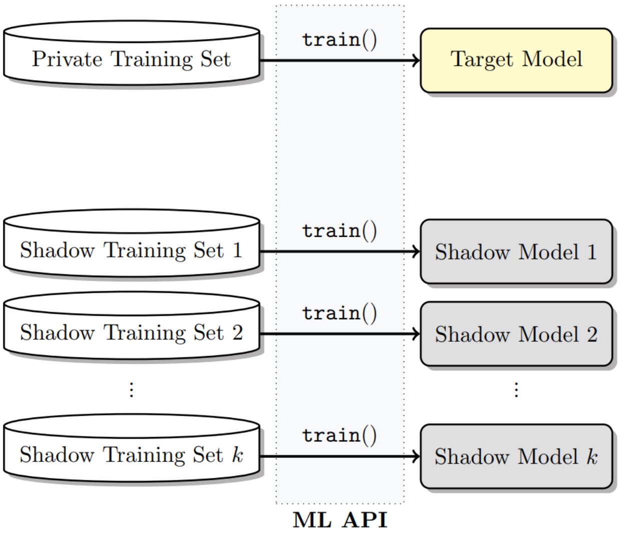

Figure 3: Overview of the training of *shadow models*adapted from [16]. *Shadow training sets* are constructed and used to train each of the models separately. The datasets share the same format but contain different data points from similar distributions.

train machine learning models that are publicly accessible. This method lets an attacker reconstruct the training data, making the potentially sensitive information available to them. The efficacy of this method is contingent upon the number and caliber of the shadow models and the congruence of their training datasets with the target model's dataset. For a detailed understanding of the specific training methodology employed by Shokri et al. for the shadow models, readers are referred to the original publication.

4.2. Knowledge Extraction Attacks with No Observable Data

In the paper titled "Knowledge Extraction with No Observable Data" [15], Yoo et al. present methods for two scenarios: available and hidden training data. NNs are parameterized functions designed to approximate arbitrary functions, specified by training data examples. The architecture of the network defines its computational structure, while the parameters or weights determine its specific computations. The paper introduces the concept of "unintended memorization", where NNs might inadvertently reveal out-of-distribution training data, termed as "secrets". From this method, the attacker extracts information stored within the NN itself. The exact nature of this information, such as the numbers depicted in Figure 5, is open to the interpretation of the attacker.

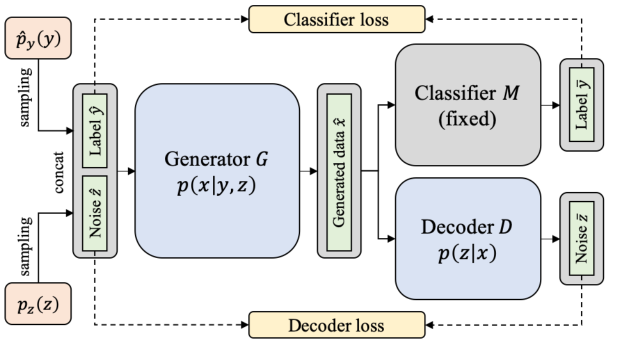

Figure 4: Overview of the KEGNET's operation adapted from [15]. The generator network G utilizes sampled variables $\hat{y}$ and $\hat{z}$ to produce a fake data point $\hat{x}$, aiming to mimic the original training data distribution by minimizing the Kullback-Leibler (KL) divergence. Concurrently, the decoder network D strives to retrieve the variable $\hat{z}$ from $\hat{x}$ and reconstruct the original input, minimizing the mean squared error (MSE) between the original and reconstructed data. Both networks undergo end-to-end training: G generates data points fed into a fixed classifier and D, while D extracts a low-dimensional representation. The iterative training refines both networks based on discrepancies between generated and original data and between original and reconstructed inputs.

For scenarios with available data, Yoo et al. [15] introduce KEGNET (Knowledge Extraction with Generative Networks). This method aims to move knowledge from a large NN (known as the *teacher network*) to a smaller one (called the *student network*). The authors designed KEGNET, especially for scenarios where there is not much training data or the student model needs to be small. Figure 4 shows a visual explanation of the KEGNET process.

However, when the original training data is concealed, especially in areas such as medicine and defense, the challenges increase. To solve this, KEGNET uses tools to create fake data points that can replace the hidden original training data. The main idea here is that the process of pulling out knowledge focuses on a small set of data points within a certain area [15]. This led to the creation of a generator network, paired with a discriminator network, to mimic and tell apart data points.

The team tested KEGNET on three datasets from the UCI Machine Learning Repository. Using a multilayer perceptron as a classifier and adding Tucker decomposition to all dense layers, the results showed KEGNET did better than other standard methods. This means an attacker can use KEGNET to pull knowledge from different NN designs and various types of training data.

While the main paper shows possible weak points in hidden models, it does not give a clear method to

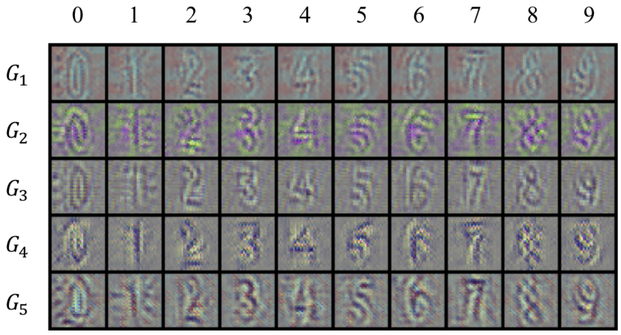

Figure 5: Visual representation of the artificial data points generated by the generator network of KEGNET [15]. These data points exemplify how the generator can produce synthetic data that closely resembles the original training data. The original teacher network was trained on the SVHN dataset [17]. Figure adapted from [15].

pull out specific details about the training data. However, the authors do mention the generator network's ability to create data points similar to the original training data. An example of this can be seen in Figure 5.

In conclusion, Yoo and his team introduce a significant advancement in machine learning by crafting a mechanism that extracts knowledge sans observable data. Their method, KEGNET, offers a solution for scenarios constrained by data accessibility due to privacy or confidentiality nuances. KEGNET provides also an opportunity for attackers to directly extract secrets stored in an NN.

5. Applications in the Networks Domain

In this section, we explore a specific application of one of the attacks (MIA) introduced in Section 4 within the networking domain. A prime example where NNs are extensively used is in Intrusion Detection Systems (IDSs).

5.1. Intrusion Detection Systems

IDSs serve as a cornerstone in the realm of network security, vigilantly monitoring network traffic to detect malicious activities [18]. A pivotal aspect of their operation hinges on the training data, which predominantly consists of network logs [18]. Figure 6 provides a visual representation of the core components and functioning of an IDS.

5.2. Learning Mechanisms of IDS

IDSs derive their efficacy from extensive training on network logs. These logs capture diverse network activities, protocols, and communication patterns. By assimilating this data, IDSs not only recognize but also learn the underlying patterns of regular and anomalous traffic. This learned knowledge empowers them to swiftly identify and respond to potential threats, ensuring robust network security [18].

5.3. Vulnerabilities and Potential Attacks on IDS

IDSs are not impervious to threats. One of the most potent threats they face is MIAs. These attacks are designed to reverse-engineer the data on which the IDS was trained. By successfully executing an MIA, attackers can gain information from the system. For instance, they can:

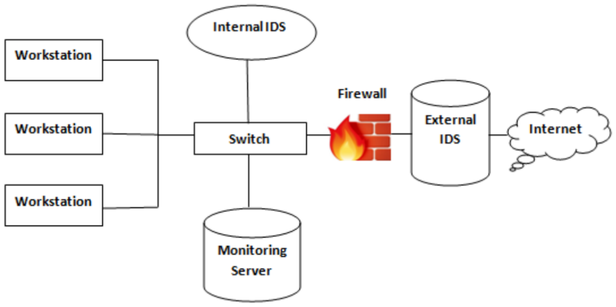

Figure 6: A schematic representation of an IDS showcasing its core components and their interactions within a network environment. This model emphasizes the critical role of IDSs in monitoring network activities, verifying connection patterns, and analyzing the flow of packets to detect potential threats. Adapted from [18].

* **Pinpoint commonly used services:** By analyzing the network logs, attackers can identify frequently used ports, such as port 53, which is typically associated with DNS.
* **Determine entities running specific services:** Through MIAs, attackers can discern which specific nodes or entities within the network are responsible for running certain services, like DNS servers.
* **Extract overarching network structure insights:** Beyond just services, MIAs can provide attackers with a broader understanding of the network's layout, its key entities, and their interrelationships.

5.4. Speculative Implications of MIAs on IDS

The implications of successful MIAs on IDS are not just limited to information extraction. Armed with the knowledge obtained from MIAs, skilled attackers can craft malicious packets that blend seamlessly with regular traffic, evading detection by the IDS. This potential scenario emphasizes the urgent need for enhanced defenses against such sophisticated attacks.
Countermeasures: While the primary focus of this paper is on the vulnerabilities, it is worth noting that the research community is not standing still. Efforts are being made to develop defense mechanisms against such attacks, as seen in [19], which proposes defense strategies based on gradient differential privacy.

6. Conclusion and future work

In this paper, we described the intricate manner in which NNs store information. Subsequently, we introduced two distinct methods for knowledge extraction from NNs when presented as a black box. In the application of one of these attacks to the networking domain, we demonstrated their significant implications, particularly within Intrusion Detection Systems.

Moving forward, more research is imperative on robust defense mechanisms, interdisciplinary collaboration between machine learning and cybersecurity, and the development of transparent and interpretable NN architectures.

Our findings underscore the importance of these areas in ensuring the security and efficacy of NNs in real-world applications.

References

[1] M. I. Jordan and T. M. Mitchell, “Machine learning: Trends, perspectives, and prospects,” *Science*, vol. 349, no. 6245, pp. 255–260, 2015. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.aaa8415

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[4] A. Graves, A.-r. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 6645–6649.

[5] W. J. von Eschenbach, “Transparency and the black box problem: Why we do not trust ai,” *Philosophy & Technology*, vol. 34, no. 4, pp. 1607–1622, 2021.

[6] M. Fredrikson, S. Jha, and T. Ristenpart, “Model inversion attacks that exploit confidence information and basic countermeasures,” in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS ’15. New York, NY, USA: Association for Computing Machinery, 2015, p. 1322–1333. [Online]. Available: https://doi.org/10.1145/2810103.2813677

[7] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, “Stealing machine learning models via prediction APIs,” in *25th USENIX Security Symposium (USENIX Security 16)*. Austin, TX: USENIX Association, Aug. 2016, pp. 601–618. [Online]. Available: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/tramer

[8] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman, “Towards the science of security and privacy in machine learning,” *arXiv preprint arXiv:1611.03814*, 2016.

[9] J. A. Anderson, *An introduction to neural networks*. MIT press, 1995.

[10] D. Svozil, V. Kvasnicka, and J. Pospichal, “Introduction to multi-layer feed-forward neural networks,” *Chemometrics and Intelligent Laboratory Systems*, vol. 39, no. 1, pp. 43–62, 1997. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0169743997000610

[11] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[12] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[13] A. Rastogi, K. Agarwal, E. Lolon, M. Mayerhofer, and O. Oduba, “Demystifying data-driven neural networks for multivariate production analysis,” in *Unconventional Resources Technology Conference, Denver, Colorado, 22-24 July 2019*. Unconventional Resources Technology Conference (URTeC); Society of . . . , 2019, pp. 2602–2622.

[14] D. Castelvecchi, “Can we open the black box of ai?” *Nature News*, vol. 538, no. 7623, p. 20, 2016.

[15] J. Yoo, M. Cho, T. Kim, and U. Kang, “Knowledge extraction with no observable data,” in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/596f713f9a7376fe90a62abaaedecc2d-Paper.pdf

[16] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, “Membership inference attacks against machine learning models,” in *2017 IEEE Symposium on Security and Privacy (SP)*, 2017, pp. 3–18.

[17] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, “Reading digits in natural images with unsupervised feature learning,” 2011.

[18] N. Unnisa A, M. Yerva, and K. M Z, “Review on intrusion detection system (ids) for network security using machine learning algorithms,” *International Research Journal on Advanced Science Hub*, vol. 4, no. 03, pp. 67–74, 2022. [Online]. Available: https://rspsciencehub.com/article_17618.html

[19] Z. Liu, R. Li, D. Miao, L. Ren, and Y. Zhao, “Membership inference defense in distributed federated learning based on gradient differential privacy and trust domain division mechanisms,” *Security and Communication Networks*, vol. 2022.