2.3 Secure Machine Learning

The field of secure machine learning explores the intersection between cyber security and machine learning. At one of the spectrum, we build learning algorithms for disambiguating patterns from noise. At the other end, cyber security desires to make patterns indistinguishable to unintended parties. Although there exists this stark dichotomy between learning and security, there is an immediate urgency for further research as deep networks increasingly pervade society.

2.3.1 Threat Models

The threat landscape is generally divided into security during training and security during inference when the fully trained models are deployed in a real system.

The first setting is concerned with how to secure training data during collaborative training. In collaborative training, multiple parties hold datasets from the same domain and agree on training towards the same objective with the same learning algorithm. For neural networks, all parties must agree on the same network topology. For example, perhaps three hospitals want to train either a single model or jointly train with their own models for classifying different types of injuries from x-ray images. Each hospital has a small dataset, but combine they have enough data that could train a highly accurate model. In the case of neural networks, each hospital training individually would only be able to train small networks. Training with a larger network would cause overfitting. A benefit of training together would enable training larger models which would allow for greater accuracy. However, to be HIPPA compliant \cite{HIPPA white papers}, the hospitals cannot aggregate all their data into a single machine, cannot transmit their data in the clear, and cannot allow the other parties to train on their data in the clear.

The second setting is concerned with how to secure the information contained in a fully training model during inference. In the past few years, there has been a slew of literature and demonstrations of attacks on real machine learning systems. Aspects of security here consider, how to design secure pipelines and systems, detecting and preventing malicious input, protection against revealing the model itself, and defending against unintended outputs.

Furthermore, these settings are studied under different assumptions of the adversary’s capabilities and behavior. Their capabilities can be of two types, white-box access and black-box access. White-box settings assume that the adversary can inspect the model parameters, make infinite modifications and queries to the model. Black-box settings, which is a subset of white-box settings, assumes the adversary cannot inspect the internals of the model, but can make some number of queries to the model and inspect the outputs. Anything that can be done in black-box attacks can also be done in white-box attacks. Thus, we can consider training under white-box access, training under black-box access, inference under white-box access, and inference under black-box access. \cite{on the protection of paper}

To further clarify the attack landscape, adversaries can be categorized by their behavior. Cryptography formally classifies adversaries as honest, honest-but-curious, semi-honest, or malicious. Thus, we can more accurately define our security assumptions as e.g. training under white-box access with semi-honest adversaries.

An even finer-grained classification of adversary capabilities and intentions consists of classification as computationally bounded or unbounded, eavesdropping or Byzantine, static or adaptive, and mobile or non-mobile.

2.3.2 Overview of Vulnerabilities

This section provides a brief summary of the current landscape of vulnerabilities in neural networks.

Model Extraction Attacks

Black-box attacks are difficult to deploy because they rely on whatever information is available. Some recently proposed defenses limit the amount of information a system provides to obfuscate how the model made its decisions e.g. hiding model confidence and limiting outputs to the top predictions \cite{goodfellow paper}. One way to circumvent this issue is to convert the problem from a black-box setting to a white-box setting by extracting the model from the system. This was first demonstrated by Tramer et.al. \cite{} in 2016. In their work they… summarize paper here

//illustrate model extraction or cite diagram from paper

Model Inversion Attacks and Membership Inference

One particular general attack on machine learning models are model inversion attacks. Given a trained model, the goal is to identify possible training points that were used to construct the model. The attack assumes the given model is discriminative and that the adversary has knowledge of the label space and knowledge of characteristics of possible samples from the inputs space. This attack can be employed in either the black-box or white-box setting. The attack works as follows. The adversary selects a valid label associated with a particular subset of samples in the input space that they would like to learn about. The adversary then generates a random vector, feeds it to the model and observes the output. If the adversary has white-box access, then the output observed are gradients and activations of neural network. If the adversary only has black-box access, the observed output is a classification or additional confidence values with the classification. The adversary then modifies the random vector to reduce the error, to move the classification closer towards the target label. This process repeats until convergence and then the adversary observes the modified input vector. If the vector does not resemble anything, then the entire process starts again with a new random vector. Alternatively, if the adversary has prior knowledge of plausible input vectors, they could initialize the vector using prior knowledge. For example, if the input domain contains images and the target label is of a particular person, the input vector could be initialized to a similar looking person or if the dataset  
<Insert illustration here>  
 Although they’re not at powerful at adversarial examples in terms the flexibility and leverage they lend to an adversary, model inversion attacks provide non-negligible information about the training set which could provide be useful for reconnaissance.

Adversarial Examples

In 2014, \cite{szgedy} discovered adversarial examples, also referred to adversarial perturbations, which are modified inputs that cause a trained neural network to misclassify their input. Further work has demonstrated that these attacks be executed in white-box and black-box settings, and work has shown successful targeted and untargeted attacks. Untargeted attacks cause the network to misclassify an input arbitrarily. Targeted attacks cause the network to classify an input as a desired class. Real-world attacks have been demonstrated and most recent work has demonstrated that three-dimensional targeted adversarial examples can be constructed to cause networks to misclassify with high error. \cite{}

Data Poisoning

In collaborative settings, training data can be corrupted to either influence the fully trained model for targeted outputs or to simply cause the network to not learn anything. The most infamous example of this type of attack occurred with Microsoft’s Tay Twitter chatbot which actively learned from interacting with twitter users. By intentionally tweeting derogatory comments to Tay, the chatbot learned the speech pattern of those users and within a day of being released learned to reply with bigoted comments causing Microsoft to remove the chatbot.

2.3.3 Overview of Defenses

This section provides a cursory overview of recently proposed defenses to the attacks mentioned above.

Differential Privacy

Homomorphic Encryption

Blockchain

Federated Learning

SMPL is a variation of federated learning