IMES DISCUSSION PAPER SERIES

Security Risks of Machine Learning Systems and Taxonomy Based on the Failure Mode Approach

Kazutoshi Kan

Discussion Paper No. 2021-E-3

IMES

INSTITUTE FOR MONETARY AND ECONOMIC STUDIES

BANK OF JAPAN

2-1-1 NIHONBASHI-HONGOKUCHO
CHUO-KU, TOKYO 103-8660
JAPAN

IMES Discussion Paper Series 2021-E-3
May 2021

Security Risks of Machine Learning Systems and Taxonomy Based on the Failure Mode Approach

Kazutoshi Kan*

Abstract

This paper clarifies the source of difficulties in machine learning security and determines the usefulness of the failure mode approach for capturing security risks of machine learning systems comprehensively. Machine learning is an inductive methodology that automatically extracts relationships among data from a huge number of input-output samples. Recently, machine learning systems have been implemented deeply in various IT systems and their social impact has been increasing. However, machine learning models have specific vulnerabilities and relevant security risks that conventional IT systems do not have. An overall picture regarding these vulnerabilities and risks has not been clarified sufficiently, and there has been no consensus about their taxonomy. Thus, this paper reveals the specificity of the security risks and describes their failure modes hierarchically by classifying them on three axes, i.e., (1) presence or absence of attacker's intention, (2) location of the vulnerabilities, and (3) functional characteristics to be lost. This paper also considers points for future utilization of machine learning in society.

Keywords: Machine learning; Failure mode; Secuirty risk; Vulnerability
JEL classification: L86, L96, M15, Z00

* Deputy Director, Institute for Monetary and Economic Studies, Bank of Japan
(E-mail: kazutoshi.kan@boj.or.jp)

The author would like to thank Fuyuki Ishikawa (National Institute of Informatics) for insightful comments. The views expressed in this paper are those of the author and do not necessarily reflect the official views of the Bank of Japan.

Table of Contents

I. Introduction

Machine learning is an inductive methodology that automatically extracts input-output relationships from a huge number of input-output samples given a predetermined model. This calculation paradigm enables us to capture extremely complicated input-output relationships and to resolve difficult tasks such as image processing. This advantage promotes the implementation of machine learning systems (i.e., IT systems that incorporate machine learning models) into social infrastructures at a rapid pace. The financial industry has also applied machine learning for their core business, e.g., asset management and credit scoring.

Nevertheless, machine learning systems are known to have various vulnerabilities. A number of them are sources of novel cyber security risks that cannot be effectively addressed by conventional security methodologies. So far, the overall picture of these vulnerabilities and relevant security risks has not been clarified, and there is no consensus about their taxonomy. Although the impacts of security risks differ among machine learning services reflecting their purposes and actual usages, those risks would cause serious incidents in a number of cases such as defeating a face recognition system or a malfunction in an autonomous driving system.

Recently, CERT Coordination Center[1] has issued a vulnerability note[2] that warned of the vulnerability in a certain class of machine learning systems. It has drawn much attention because it was the first alert regarding machine learning that is expected to be implemented more widely and deeply into IT systems in the future. The alert was novel in a sense that the vulnerability pointed out was not of an individual and existing IT system but of a certain class of machine learning systems including ones that do not exist yet. We need to be aware of known vulnerabilities of machine learning systems when we utilize them.

In this paper, Section II explains the paradigm and features of machine learning. Section III points out the cyber security risks specific to machine learning systems. It also clarifies sources of the difficulties in machine learning security and find the usefulness of the taxonomy of security risks based on the

[1] CERT Coordination Center is a non-profit organization centered at Carnegie Mellon University in the United States. It conducts research on cyber security, collects information on vulnerabilities mainly in existing cyber systems, and shares them with software vendors and incidents responders. Through these networking activities, it warns people of cyber security risks and promotes to resolve them.

[2] Vulnerability Note VU#425163, March 19, 2020 (https://www.kb.cert.org/vuls/id/425163).

failure mode approach proposed by Kumar *et al.* (2019). Referring to this literature, Sections IV, V, and VI introduce known vulnerabilities of machine learning systems. Kumar *et al.* (2019) collected research results on 'failures' of machine learning systems, classified them by failure modes, and suggested a comprehensive taxonomy on the basis of the failure mode approach. Their paper is cited by the aforementioned vulnerability note from CERT Coordination Center.

II. Paradigm and features of machine learning

A. Comparison between ordinary IT systems and machine learning systems

1. Ordinary IT systems

In an ordinary IT system (i.e., an IT system that does not incorporate machine learning), the input-output relationship (f) is predetermined as its specification (see the top of Figure 1). System developers implement the information processing rules of the system in accordance with this specification. By construction, those rules do not depend on any data. Thus, the ordinary system is deductive because it derives output data from input data in accordance with predetermined information processing rules.

2. Machine learning systems

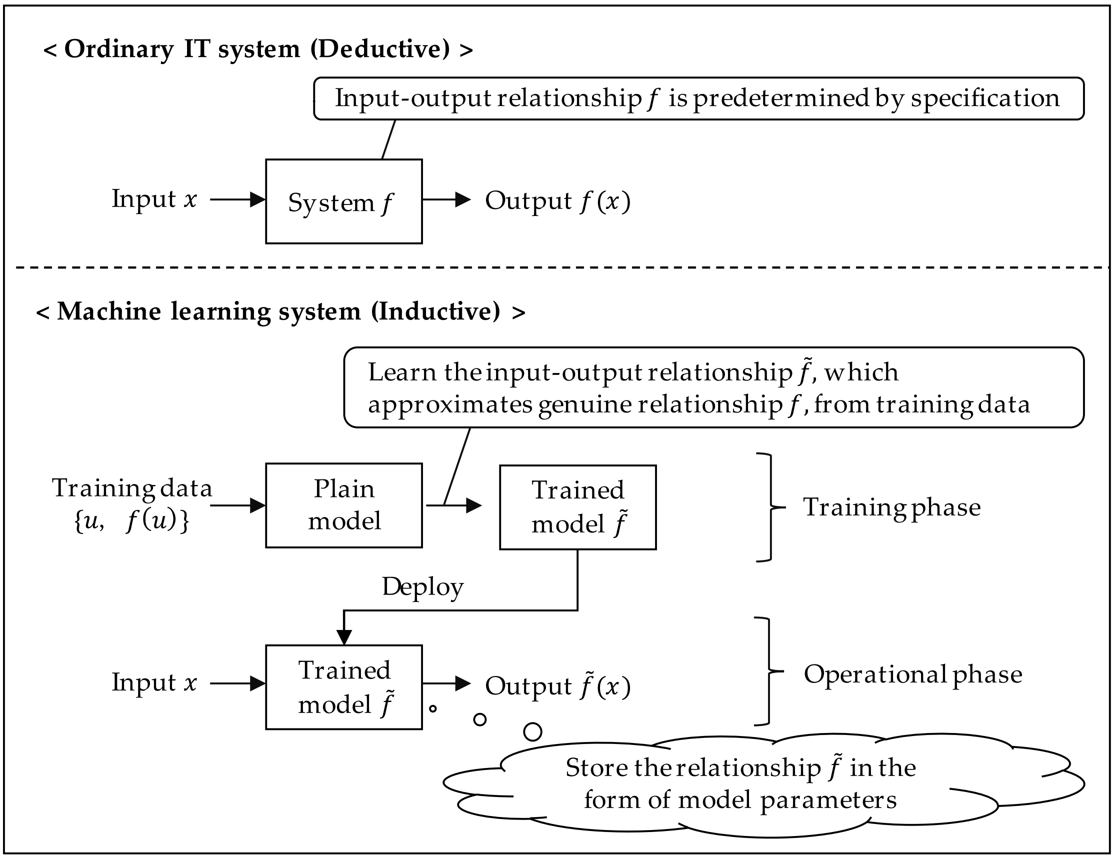
In a machine learning system (see the bottom of Figure 1), a machine learning model automatically extracts the input-output relationship ($\tilde{f}$) from training data $\{u, f(u)\}$, which consists of input-output samples.[3] This process is called

[3] Figure 1 shows the paradigm of supervised learning in which machine learning models are trained using samples of input-output pairs $\{u, f(u)\}$. In general, there are three machine learning paradigms: supervised learning, unsupervised learning, and reinforcement learning. All share a common property in which the information processing rules depend on (training) data.

In unsupervised learning, the training data does not contain samples of correct output $f(u)$. Its typical applications are clustering, which automatically classifies unknown data, and anomaly detection, which finds outliers in data.

In reinforcement learning, machine learning models produce data for training by themselves given an environment. Reinforcement learning consists of agents, rewards, and an environment. The agent repeatedly selects and takes an action, and receives a 'reward' or 'gain' from the environment, which are fed back into the agent. Through trial and error, the agent learns a better strategy of selecting actions. This methodology is distinguished from other paradigms due to its characteristics that it does not necessarily need a huge amount of (training) data. For instance,

Figure 1. Comparison of ordinary IT system and machine learning system


< Ordinary IT system (Deductive) >
Input-output relationship f is predetermined by specification
Input x
System f
Output $f(x)$
< Machine learning system (Inductive) >
Learn the input-output relationship $\tilde{f}$, which approximates genuine relationship f, from training data
Training data $\{u,\ f(u)\}$
Plain model
Trained model $\tilde{f}$
Training phase
Deploy
Input x
Trained model $\tilde{f}$
Output $\tilde{f}(x)$
Operational phase
Store the relationship $\tilde{f}$ in the form of model parameters

'training' or 'learning.' The extracted relationship ($\tilde{f}$) approximates the genuine and unobservable one (f), which appears only in the training data, and also represents the information processing rules of the system. This means that preparing for the training data is equivalent to formulating the specification in the case of an ordinary IT system. System developers cannot directly affect the rules because they depend only on training data and plain machine learning models.[4] Thus, a machine learning system is inductive because it derives information processing rules from individual data.

To realize and implement the learning mechanism, machine learning

reinforcement learning is applied to artificial intelligent software for playing Go, a board game, and for autonomous walking robots.

[4] System developers can indirectly affect the information processing rules of a machine learning system through the construction of training data and the choice of learning algorithms and plain models.

methodology requires an approach of preparing flexible models[5] that are suitable for general purposes. These models can express various and complicated input-output relationships flexibly by modifying their parameters. Model training corresponds to determining the values of these parameters from training data by running a learning algorithm. This stage of information processing is called the 'training phase.' The more expressive the model needs to be, the more parameters (degrees of freedom) the model needs to contain. The trained model possesses the information processing rules in the form of the determined values. The stage in which the trained model incorporated into the system provides a service is called the 'operational phase.'

B. Features of machine learning

Machine learning enables us to obtain plausible output from a huge amount of training data, even if the desirable input-output relationship is unknown or too complicated to be expressed as information processing rules suitable for coding. In contrast, machine learning has the following disadvantages that lead to difficulties in cyber security and software quality management.

1. Unclear requirements for machine learning systems

Machine learning systems are expected to resolve tasks where input data come directly from natural or real-world environments. Since the range of input data is vague and open, the properties that the input-output relationship of the system should satisfy are also vague. In other words, we cannot accurately define requirements for machine learning systems. For example, an automatic driving system that uses image processing is expected to output appropriate operations of a vehicle for any situations a driver may encounter. In this case, the input data will vary indefinitely, reflecting real-world factors such as road traffic conditions, weather, and human behavior. It is impossible to enumerate all of them, and thus impossible to define the range of input data needed to determine the behavior of the system. This also indicates that it is infeasible to completely validate all input-output relationships in the system.

[5] Various models have been proposed that can be applied to typical tasks in machine learning: regression, prediction, classification, and anomaly detection. Frequently-used examples include the multilayer perceptron (MLP), which is the simplest version of deep learning, and the random forest model, which bundles a multitude of decision trees for a single output.

2. Unclear features of trained models

Expressive machine learning models are typically required to resolve difficult tasks of which the requirements are unclear. Since information processing rules are stored in a model as a huge number of parameters in this case, the machine learning system has the following three features regarding explainability, predictability and confidentiality, respectively.[6] First, it is difficult for humans to interpret model parameters as meaningful information processing rules, making it difficult to justify a model's performance. Second, it is difficult to predict in advance how the model will perform for input data that do not appear in the training data. Third, the model-training process conveys information from training data into model parameters, but information conserved by this conveyance is unclear. As I discuss later, this feature poses a risk of information leakage.

III. Security risks specific to machine learning systems

A. Vulnerabilities inherent in information processing rules

A number of vulnerabilities in machine learning systems can be inherent in information processing rules (a set of parameters in a machine learning model).[7] In fact, many studies have discovered feasible attacks that exploit these types of vulnerabilities (Kumar *et al*. [2019]).

For example, Sharif *et al*. (2017) suggested an attack against a machine learning system that identifies a person from a face image captured by a camera. When a user (an attacker) wears accessories such as glasses whose surface is maliciously designed, the system cannot correctly recognize the attacker as the genuine person due to changes in the input image caused by the glasses. This attack invalidates the identification by exploiting a deficiency in the information processing rules acquired by the model through its training phase.

In this example, even though the attacker can only exercise the privileges permissible to them (and this privileges must remain permissible from the perspective of the system design) and there are no conventional software bugs, the attack is considered successful. This is a novel security risk brought about by machine learning. It is difficult to effectively resolve this risk with only

[6] It is also unclear how these features relate to each other.

[7] There are also vulnerabilities common to machine learning systems and ordinary IT systems. For example, the inappropriate assignment of user privileges or programming errors (software bugs). For most of them, however, the countermeasures have already been established.

conventional security countermeasures such as fixing software bugs, controlling access, and managing user privileges appropriately.

B. Source of difficulties in machine learning security

Resolving vulnerabilities inherent in information processing rules of machine learning are not easy due to the following three difficulties.

1. Identifying vulnerabilities comprehensively

The first difficulty lies in comprehensively identifying vulnerabilities of machine learning systems. This is because, as mentioned in Section II.B.1, system requirements are unclear and it is difficult to exhaustively take into account all possible input data and their corresponding plausible output data. Although research has been conducted actively on the vulnerabilities and associated security risks, an overall picture of them has not been clarified and there is no consensus about their taxonomy. For system developers, it is also costly in terms of time and human resources to follow the latest research trends timely and thoroughly.

2. Modifying information processing rules

The second difficulty lies in modifying the information processing rules (i.e., retraining a model), predicting potential impacts of the modification on the performance of the model, and validating the modification. Due to the poor explainability and predictability as mentioned in Section II.B.2, it is difficult to find a way to remove vulnerabilities inherent in the information processing rules from the model parameters. It is also not easy to predict impacts of the modification on the system functionalities. Furthermore, it is difficult to validate that modified rules to satisfy the system requirements because they are unclear, as discussed. As a result, it is difficult to modify the rules to appropriately mitigate or resolve vulnerabilities inherent to them. Thus, there are always uncertainties in conducting such security measures that change information processing rules of a model.[8]

3. Separation of vulnerabilities and functionality

Modifying information processing rules to mitigate vulnerabilities affects the

[8] Imperfection in machine learning security can be viewed as that of the performance of machine learning systems in the context of software quality management.

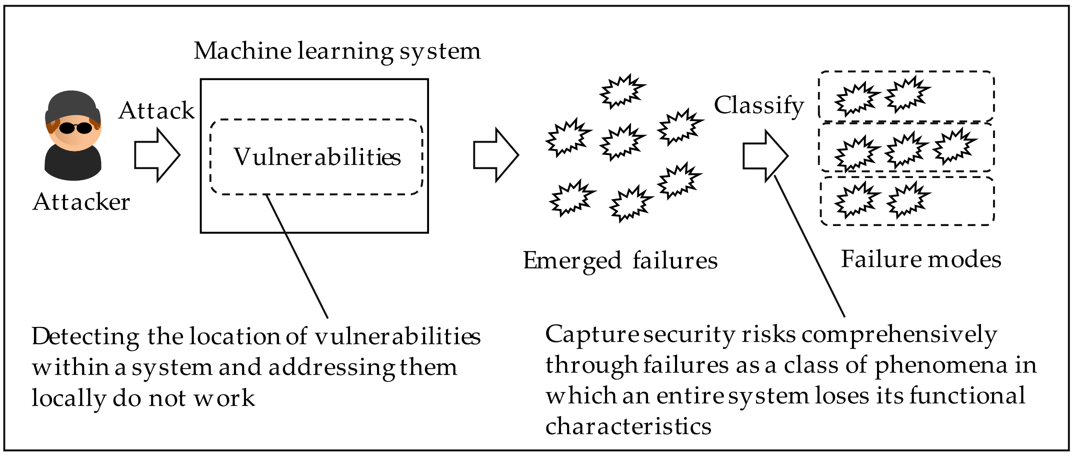
functionality of the system. When retraining a model, the overall parameters are changed. Even if only a part of the parameters is changed, its impact will extend to the functionality of the entire system since it is materialized by a set of all the parameters. Therefore, the vulnerabilities inherent in the information processing rules and the functionality cannot be considered separately. The issue is how to successfully modify the rules while maintaining the functionality.[9]

IV. Failure modes in machine learning: Approach of Kumar *et al.*

A. Failure mode

Vulnerabilities inherent in the information processing rules of machine learning systems cannot be addressed without modifying their entire functionality, as discussed in Section III. Thus, security risks associated with the vulnerabilities can be captured only by a comprehensive approach for the entire system, rather than a partial approach that reveals the location of each vulnerability within the system and addresses each of them locally. Kumar *et al.* (2019) adopted the comprehensive approach that focused on the 'failure' phenomenon of an entire system in which the system loses its characteristics or attributes to be retained. They collected results related to failures in machine learning systems and

Figure 2. Failure modes in machine learning systems


Machine learning system
Attack
Attacker
Vulnerabilities
Classify
Emerged failures
Failure modes
Detecting the location of vulnerabilities within a system and addressing them locally do not work
Capture security risks comprehensively through failures as a class of phenomena in which an entire system loses its functional characteristics

Note: Failures can occur without attacks.

[9] The difficulty in capturing security risks of machine learning systems is also reflected in the vulnerability note published by CERT Coordination Center (see Footnote 2). The report states that machine learning models trained using the gradient descent method can be forced to make arbitrary misclassifications by an attacker that can influence the items to be classified. The note points out that such vulnerabilities come from learning algorithms themselves.

classified them by 'failure modes' (see Figure 2). A list of the modes proposes a taxonomy of vulnerabilities and the associated security risks in machine learning.[10]

B. Classification of failure mode

Failure modes listed by Kumar *et al.* (2019) have the following three attributes that are helpful to structure them (see Figure 3). Kumar *et al.* (2019) have already proposed to attach the first and the third attributes to each failure mode. I simply extend their work in such a way to add the second attribute here.

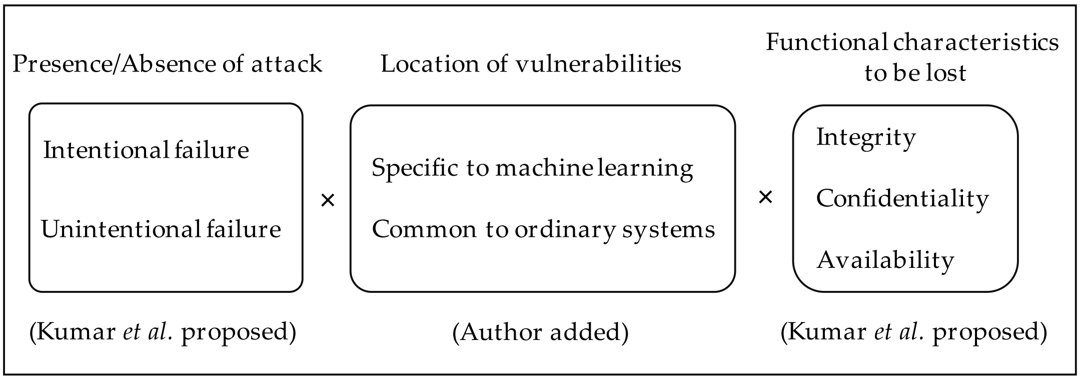
1. Presence or absence of attacker's intention

Failures modes in machine learning can be classified into two categories: 'intentional failures' caused by an attacker's malicious intention and 'unintentional failures' that originate from the innate software design of the system.

Intentional failures consist of three types of attacks: (1) attacks that manipulate a model and obtain incorrect output data in such a way to perturb input data, (2) attacks that change the information processing rules in such a way to set a backdoor in a model by poisoning training data, and (3) attacks that steal information such as detecting hidden training data or duplicating the model itself with business value.

Unintentional failures focus on cases where the performance of the model autonomously degrades during the training process without external

Figure 3. Attributes for classifying failure modes


Presence/Absence of attack
Location of vulnerabilities
Functional characteristics to be lost
Intentional failure
Unintentional failure
×
Specific to machine learning
Common to ordinary systems
×
Integrity
Confidentiality
Availability
(Kumar et al. proposed)
(Author added)
(Kumar et al. proposed)

[10] The list of failure modes and research cases shown in Kumar *et al.* (2019) are still being updated. The list is useful to understand the latest research trends. This article is based on a report obtained at the time of writing (end of July 2020).

interference such as attacks. This category excludes failures due to negligence in the training process or simple software bugs. Thus, there are many cases of failures in reinforcement learning in which training data are autonomously generated through the trial and error of agents within the environment, and it does not need a huge amount of training data from the outside.

2. Location of vulnerability

Failures of machine learning systems can be conventionally classified into the following two types in accordance with the location of vulnerabilities: failures caused only by vulnerabilities specific to machine learning systems as discussed in Section III and failures caused by vulnerabilities common to ordinary systems. Recently, Machine Learning as a Service (MLaaS), which provides a platform for machine learning development via the Internet, has become commonly used. Since such large-scale services are usually provided using open-source software and/or open-access databases, conventional software bugs or malicious codes can be mixed into the machine learning model developed on the platform. Though these vulnerabilities are not specific to machine learning, we cannot ignore them when securely implementing machine learning systems. In particular, it should be noted that the combination of these conventional vulnerabilities and those specific to machine learning enables new types of attacks and associated failure modes.

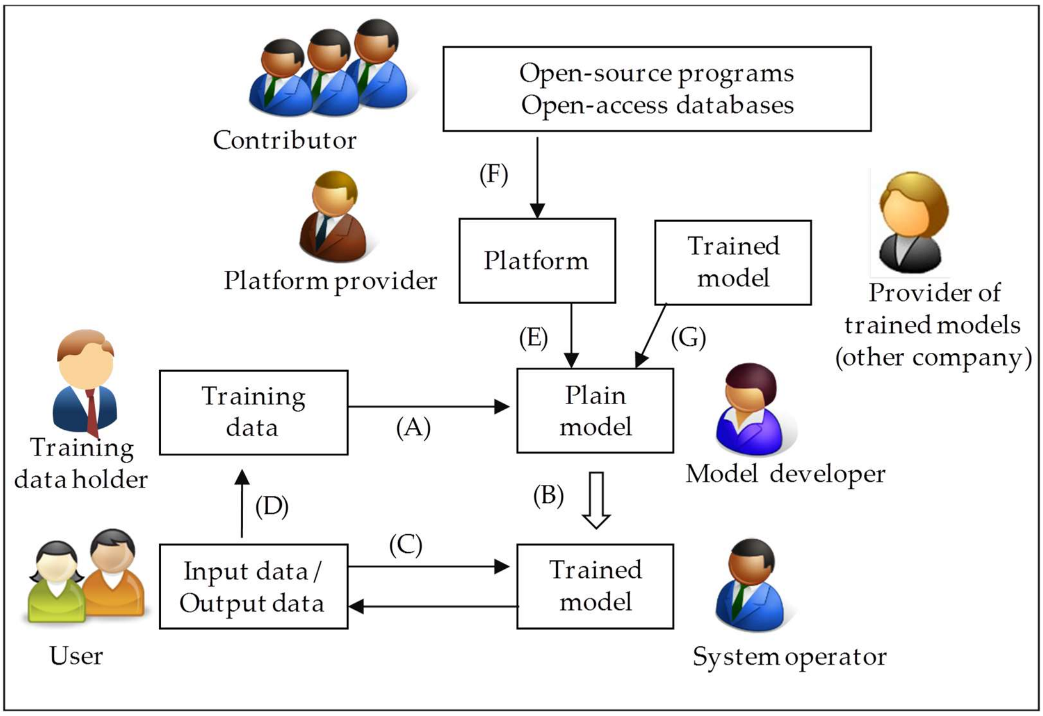
3. Functional characteristics to be lost

Generally, the security principle of an IT system is to retain confidentiality, integrity, and availability (CIA). Confidentiality represents a characteristic where operation data are not disclosed to unauthorized entities. Integrity represents one where an IT system operates as specified. Availability represents one where an IT system always operates in response to user requests. A successful attack can violate any of these characteristics. Therefore, it is natural to categorize failures in accordance with the characteristics to be lost due to such an attack. This conceptual framework for ordinary IT systems is applicable to the security analysis of machine learning systems.

V. Diagram of information flows in providing machine learning systems

As we will see in Section VI, many failures in machine learning are caused by attacks. To recognize how these attacks are actually carried out, it is useful to

Figure 4. Information flows among entities in machine learning systems



overview the flows of information, e.g., models, data, and program source codes, regarding the development of machine learning systems. Figure 4 visualizes such flows.[11]

The simplest scheme of providing a machine learning system is as follows. In the training phase, (A) a model developer receives training data from a training data holder and trains the model. (B) The trained model is deployed in the production environment. In the operational phase, (C) a system operator provides users with machine learning services in response to users' demand. (D) New data, which are generated in the operation of the services, are stored as new training data.

In terms of model development, (E) the model developer may use an

[11] In this article, we do not individually associate each type of attack seen in Section VI with situations in which they are executed (locations in the diagram in Figure 4). There are considered to be multiple such situations. For example, the data poisoning attack (see Section VI.A.1.b.(1)) can be executed in each process of generating, storing, and distributing training data (A, D, E, F, etc. in Figure 4). Our future work is to enumerate conditions that enables each attack (premise of attacker's ability, position of an attacker, situation where an attack is executed) and clarify security countermeasures under each condition.

external platform. (F) Typically, a large-scale platform for machine learning is developed using open-source programs or open-access databases. (G) The model developer may import a trained model (from other companies), which typically needs large-scale computational resources for its training, and incorporates it into their own model as a part.[12]

VI. List of failure modes by Kumar *et al.*

This section introduces a list of failure modes illustrated by Kumar *et al.* (2019) in a hierarchized way. They classified failure modes into 'intentional failures' (11 modes) and 'unintentional failures' (6 modes). I further classified the intentional failures into attacks that exploit vulnerabilities purely specific to machine learning (7 modes) and attacks that additionally exploit conventional vulnerabilities (4 modes). These two classes are further classified in accordance with security characteristics to be lost, i.e., integrity, confidentiality, and availability.

Classification of failure modes

A. Intentional failures (11 modes)
  1. Attacks exploiting vulnerabilities purely specific to machine learning (7 modes)
    a. Attack on integrity 1 <Changing input data>
    b. Attack on integrity 2 <Modifying information processing rules>
    c. Attack on confidentiality
  2. Attacks exploiting additional common vulnerabilities (4 modes)
    a. Attack on integrity
    b. Attack on confidentiality
    c. Attack on integrity, confidentiality, and availability
B. Unintentional failures (6 modes)

[12] Transfer learning is included in these cases. It is a technique for applying a trained model that solves one problem to another related problem. It is attractive because it opens the way to apply a methodology, which requires a large amount of training data such as deep learning, to problems when only a small amount of training data is available. For example, a deep learning model for object recognition trained and published by a company is incorporated into a model fine-tuned by another company for their own purpose. For the vulnerability of transfer learning, see Section VI.A.2.a.(1) (Setting a backdoor).

A. Intentional failures

Most studies on intentional failures aim to prove the existence of vulnerabilities in machine learning systems and do not necessarily indicate the magnitude of threats or levels of risks[13] that vulnerabilities will be exploited in actual situations. To evaluate the security risks of a machine learning system in each failure mode, it is also necessary to consider the system's purpose, its operating environment, and assumptions[14] about an attacker's ability depending on the attack method.

1. Attacks exploiting vulnerabilities purely specific to machine learning

There are three types of attacks that exploit vulnerabilities purely specific to machine learning: (1) those that attempt to manipulate output by changing input data fed into the trained model without changing the information processing rules (Attack on integrity 1), (2) those that attempt to modify the rules themselves maliciously (Attack on integrity 2), and (3) those that attempt to steal information from the machine learning model (Attack on confidentiality).

a. Attack on integrity 1 <Changing input data>

Attacks that attempt to manipulate the output by changing the input data include perturbation attacks, making adversarial examples in physical domains. The following sections summarize each failure mode and its corresponding research cases. We also describe ‘attack modes’ that represent the author's presumption of how an attacker exploits the vulnerabilities in each failure mode (the same applies hereinafter).

(1) Perturbation attack

Failure mode	The output of a machine learning model is manipulated improperly by modifying the input data.

[13] Microsoft’s supplemental content named ‘Bug Bar’ describes each intentional failure mode and puts ratings of severity for it (https://docs.microsoft.com/en-us/security/engineering/bug-bar-aiml).

[14] This paper only describes the situation of each attack and does not detail the premise of the attacker's ability. In the case of a white-box attack, the attacker has full knowledge of a target model such as model parameters and architecture. In the case of a black-box attack, the attacker knows nothing about the internal state of the model. In general, the premise of the attacker’s ability depends on the attack method. For example, perturbation attacks (see Section VI.A.1.a.(1)) contain both black-box and white-box attacks.

Attack mode	An attacker perturbs the input data of a query to the machine learning system to obtain a desired output.

<Research cases>

- An attacker constructs an adversarial image by adding noise, which is imperceptible to human eyes, to an X-ray image of the skin. The crafted adversarial image is perceptually indistinguishable from the original one but causes the skin lesion classifier, which is based on a deep learning model, to misclassify. Similarly, adding noise to MRI images of the whole brain prevents the correct segmentation maps from being generated (Paschali *et al.* [2018]).
- An attacker performs advanced editing operations on text translation systems, which are based on a deep learning model, by slightly modifying the characters in the input text data. The operations can remove or change a specific word from the translated text (Ebrahimi, Lowd, and Dou [2018]).
- An attacker constructs adversarial audio examples on automatic speech recognition systems, which are based on a deep learning model. The adversarial waveform is quite similar to the original one, but transcribes into any phrase chosen by the attacker (Carlini and Wagner [2018]).

(2) Adversarial example in the physical domain

Failure mode	A machine learning system malfunctions due to input data associated with maliciously crafted adversarial objects in the physical domain.
Attack mode	An attacker creates an object that deceives the machine learning system and places it in a specific location.

<Research cases>

- An attacker constructs robust adversarial 3D objects with custom textures. The image recognition system consistently misidentifies images of turtle-shaped objects as those of rifles, regardless of viewpoint shifts or other natural transformations (Athalye *et al.* [2018]). This attack can confuse the system for detecting dangerous goods.

- An attacker constructs adversarial sunglasses with custom and inconspicuous textures that can fool an image recognition system. The system can no longer identify the person wearing sunglasses correctly (Sharif et al. [2017]). This attack also enables the person to evade manual face recognition by humans.

b. Attack on integrity 2 <Modifying information processing rules>
Attacks that attempt to change the information processing rules maliciously include data poisoning attacks and adversarial reprogramming.

(1) Data poisoning attack

Failure mode	Information processing rules are maliciously modified by contaminating the training data.
Attack mode	An attacker injects improper data into the training data managed by the training data holder. Alternatively, in an online machine learning system, the attacker generates improper data with user privileges, which is then joined with the training data.

<Research cases>

- A certain chatbot had adopted an online learning methodology in which conversation logs with users were fed back into the system as training data, enabling the chatbot to acquire inappropriate expressions through conversations with multiple malicious users (Lee [2016]).
- An attacker introduced malicious samples into the training data with an 8% poisoning rate for the model that predicts the dosage of an anticoagulant drug. The prediction of the model based on LASSO regression dramatically changed by 75% for half of all patients (Jagielski *et al.* [2018]).

(2) Adversarial reprogramming

Failure	A model, which is trained to perform original task X, is

mode	maliciously used for another task Y chosen by the attacker without retraining the model.
Attack mode	Given the trained model designed originally to perform X, an attacker crafts the converter (an adversarial program) f that converts Y's input so that the model performs Y for the converted input. In advance of the craft, the attacker defines another converter g that maps the model's output back to Y's output. During the operational phase, the attacker converts Y's input I_Y into the model's input $f(I_Y)$ using the adversarial program, and puts $f(I_Y)$ into the model. The attacker then maps the model's output $M \circ f(I_Y)$ back to Y's output $O_Y = g \circ M \circ f(I_Y)$. Through this process, the attacker can enable the model to perform Y different from X.

<Research case>

➢ Given a pre-trained ImageNet model that classifies images, an attacker crafts an adversarial program that enables the model to do a counting task. The program places multiple small rectangles at the center of a specific image. The ImageNet model then takes the converted image including small rectangles as an input. The model outputs its 'classification' that actually depends on the number of rectangles centered in the input image. Mapping the 'classification' into the corresponding number of rectangles, the attacker can enable the model to perform the counting task as desired (Elsayed, Goodfellow, and Sohl-Dickstein [2018]]).[15]

c. Attack on confidentiality

Attacks that attempt to extract information stealthily from machine learning

[15] This research aims to explore the potential of attacks against deep neural networks. The severity of the threats from those attacks in actual situations has not been sufficiently studied. Elsayed, Goodfellow, and Sohl-Dickstein (2018) reported the risk that computational resources are stolen and reprogrammed for malicious and unethical usages such as breaking a CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart), which distinguishes between machine and humans input.

models include model inversion, membership inference, and model stealing.

(1) Model inversion

Failure mode	Secret training data or hidden features used in a machine learning model are recovered from its output through carefully designed queries.
Attack mode	An attacker has a part of the training data (such as a user's name in the database of a face recognition system) and repeatedly accesses the model as an ordinary user.

<Research case>

- An attacker reconstructs a private and recognizable face image from a corresponding user's name and output data from the face recognition model.[16] This attack utilizes the confidence values that are included in the output of the model. In another example, the attacker infers the individual responses to sensitive questions such as ``Have you ever cheated on your significant other?" with high precision. This phenomenon occurred in a certain decision tree model that studied lifestyle surveys (Fredrikson, Jha, and Ristenpart [2015]).

(2) Membership inference attack

Failure mode	Whether a training dataset contains a given data record or not is determined.
Attack mode	An attacker has a candidate for the data record and repeatedly accesses the machine learning model as an ordinary user.

<Research case>

- A target model is trained using a hospital discharge dataset that

[16] To quantify the efficacy of reconstructing face images, the researchers performed an experiment to determine whether humans can correctly select a target person out of five images by using the recovered one. As a result, participants of the experiment selected the correct images at almost 95% accuracy.

contains sensitive attributes regarding a patient such as diagnoses, procedures underwent (e.g., surgery), and generic information (e.g., gender, age, hospital id). The attacker predicts the patient's procedure on the basis of the attributes with over 70% accuracy (Shokri *et al*. [2017]).

(3) Model stealing

Failure mode	An original model and its functionality are recreated.
Attack mode	An attacker repeatedly sends queries to the target model (i.e., the original model) as a user to recreate the new one that performs similarly to the original one.

<Research case>

➢ An attacker duplicates the functionality of a target machine learning model (i.e., steals the model) without prior knowledge of its parameters or training data. For example, a decision tree model, which was trained using a credit database and outputs credit scores, was duplicated by the model stealing attack. This attack was found to be effective for the machine learning services accessible over a network (Tramèr *et al.* [2016]).

2. Attacks exploiting additional common vulnerabilities

Attacks that exploit common vulnerabilities include (1) backdoor attacks exploiting the supply chain of machine learning models (Attack on integrity), (2) recovering training data (Attack on confidentiality), and (3) exploiting software dependencies (Attack on integrity, confidentiality, and availability).

a. Attack on integrity

(1) Backdoor attack[17]

Failure mode	A pre-trained model with a backdoor is provided by a malicious third party. The model performs normally in the absence of a trigger secretly known to an attacker, but

[17] Gao *et al.* (2020) surveyed the backdoor attacks and suggested the taxonomy of them.

	behaves maliciously when the trigger is stamped with the input.
Attack mode	In a situation in which the training of a model is outsourced[18], an attacker (the third party model provider) provides a backdoored model to the client (the model developer).

<Research cases>

- Gu, Dolan-Gavitt, and Garg (2019) demonstrated the backdoor attack. An attacker trains an image recognition system with a backdoor for a traffic sign detection task. The trigger is chosen by the attacker to be a small yellow square attached to the bottom of the traffic sign. The model misclassified more than 90% of the stop signs with the trigger as speed-limit signs, as aimed by the attacker. Moreover, the model recognized the clean traffic signs (without the trigger) with less than a 1% drop in accuracy compared to the baseline model without the backdoor. Thus, the backdoored model is mostly indistinguishable from the normal one for people without prior knowledge of the trigger.

  The backdoor can be maintained during transfer learning. The backdoored model retrained for detecting Swedish traffic signs shows a 25% drop in accuracy on average when the trigger is present.

- Liu *et al*. (2017) [19] suggested another backdoor attack effective in situations where an attacker has full access to the model but not to the training data. The attacker generates a small set of training data in accordance with the trigger, and retrains the model by modifying a part of its parameters. The modified model malfunctions for the input data stamped with the trigger. When applied to a face recognition model, it misclassifies an arbitrary person as a specific person for the input image stamped with a small square (i.e., the trigger). This attack can be applied to a wide range of models such

[18] For example, the training of deep learning models for object recognition requires a huge amount of computational power. Thus, model developers who lack computational resources have an incentive to outsource the training procedure to a third party.

[19] This article is not cited in Kumar *et al.* (2019).

as a speech recognition or autonomous driving model.

(2) Attack to the supply chain of machine learning models

Failure mode	Trained models provided by repositories [20] are contaminated. When such models are invalid, the retrained or reused model (using transfer learning) performs incorrectly.
Attack mode	An attacker puts a malicious model into the repository or replaces an existing model with a tampered one. Alternatively, the attacker intrudes upon the repository server and enables it to distribute the malicious model.

<Research case>

- Backdoored models can be distributed from repositories that provide pre-trained ones (e.g., Caffe Model Zoo), and can also be used in open-source machine learning frameworks (e.g., TensorFlow, Keras, Core ML, Theano, MXNet) by utilizing the conversion scripts. By exploiting these supply chains, an attacker can contaminate a large number of models. In fact, the proclaimed hash value for a pre-trained model hosted by Caffe Model Zoo does not match that of its downloaded version in a number of cases (Gu, Dolan-Gavitt, and Garg [2019]).

b. Attack on confidentiality: Stealing the training data

Failure mode	A malicious platform provider recovers private training data from the output of a customer's model through queries.
Attack mode	If a customer develops a model on a platform powered by the malicious provider, the provider can recover the training data solely from the output of the model by feeding queries that run a backdoored algorithm.

c. Attack on integrity, confidentiality, and availability: Exploiting software

[20] A repository stores and provides digital content such as software source code or models.

dependencies and conventional bugs

Failure mode	Conventional bugs that exist in a machine learning system cause it to crash or to be manipulated.
Attack mode	An attacker finds common software bugs in the basic programs (e.g., the numerical calculation library) that underlie the system or embeds bugs into the system intentionally. The bugs are exploited to achieve its malicious goal (e.g., to crash the system, or to skip verification operations by the system).

<Research case>

- An attacker can perform a denial-of-service (DoS) attack, which prevents the system to operate a service, by exploiting bugs that cause the system to crash, enter an infinite loop, or exhaust all memory. The attacker can also perform an evasion attack, which induces misclassification, by overwriting the output of the model or hijacking the application control flow (Xiao et al. [2018]).

B. Unintentional failures

Unintentional failures include (1) reward hacking, (2) side effects, (3) distributional shifts of input data, (4) natural adversarial examples, (5) common corruption, and (6) incomplete testing for a realistic environment.

1. Reward hacking

Failure mode	A reinforcement learning system performs in unintended ways because of a mismatch between implemented and ideal rewards.[21]

2. Side effect

Failure mode	A reinforcement learning system has a disruptive effect on an external environment while achieving its purpose.

[21] Here is a list of games that incorporate artificial intelligence such as reinforcement learning systems (https://docs.google.com/spreadsheets/d/e/2PACX-1vRPiprOaC3HsCf5Tuum8bRfzYUiKLRqJmbOoC-32JorNdfyTiRRsR7Ea5eWtvsWzuxo8bjOxCG84dAg/pubhtml).

<Research case>

- In a certain scenario, a robot knocks over a vase filled with water on the path to convey an object (Amodei *et al*. [2016]).

3. Distributional shift in input data

Failure mode	A machine learning system performs unstably because it cannot adapt to the changes in the probability distribution of input data in the test environment from that in the training one.

<Research case>

- Leike *et al*. (2017) trained two state-of-the-art reinforcement learning agents, A2C and Rainbow DQN, in a simulation environment to avoid a lava lake while moving from the start to the goal. The agents avoided the lava successfully in the training environment, but failed to do so robustly in the test environment where the distribution of the lava slightly changed. This occurred because the training data did not contain sufficient variants of the distributional shifts.

4. Natural adversarial examples

Failure mode	A machine learning model misrecognizes input samples that are found in the real world.

<Research case>

- Adversarial examples can be found in the real world (Gilmer *et al*. [2018]). Hard Example Mining (HEM) automatically selects hard examples that are difficult for machine learning models to recognize properly, and trains the models focusing on those examples to improve accuracy and reduce the cost of training. Models can be confused by taking the adversarial instances sampled by HEM as input.

5. Common corruption

Failure mode	The performance of a machine learning model deteriorates due to common corruptions or perturbations that frequently occur in natural situations or usages.

<Research case>

- Hendrycks and Dietterich (2019) established benchmarks for image classifier robustness to common perturbations (brightness, contrast, blur, weather, noise, etc.) that reduce the accuracy of image classifiers.

6. Incomplete testing in realistic conditions

Failure mode	A machine learning model is not able to perform well in operational environments due to insufficient performance tests under realistic conditions.

<Research case>

- Gilmer *et al*. (2018) reported that robust machine learning models sometimes fail to perform well in natural and realistic environments. Examples of misclassifications include "stop" signs blown over in the wind and pedestrians wearing shirts with traffic signs printed on them, which should be recognized correctly by the automatic driving system.

VII. Conclusion and implications for machine learning security

An overall picture of the vulnerabilities and associated security risks of machine learning systems has not been completely clarified. Thus, organizations that provide services using machine learning, including financial institutions, should continue to collect information on the vulnerabilities and evaluate each security risk in light of their own purposes and environment in order to implement appropriate security countermeasures. This section presents key points in machine learning security and the usefulness of the failure mode approach.

A. Follow the latest research on machine learning security

Understanding the vulnerabilities is necessary to develop cyber security strategies. In the case that the information processing rules themselves incorporate vulnerabilities in a certain machine learning system, enabling users' unrestricted access to the model can lead to integrity or confidentiality violations. Moreover, it is generally difficult to detect attacks that exploit vulnerabilities specific to machine learning (such as the data poisoning) after such attacks are executed. Countermeasures for these issues have not yet been stylized in a comprehensive and organized manner.[22] The list of failure modes will be helpful to learn the characteristics of security risks and to derive insightful implications for cyber security. As long as the list in Kumar *et al*. (2019) remains open-ended and is updated, it will enable cyber security experts to reduce their workload to follow the latest research by themselves in a timely manner.

B. Security risk regarding the supply chain

Vulnerabilities could be incorporated into machine learning systems through the supply chain. It becomes more difficult to develop the whole system in-house due to resource restrictions of individual institutions. Vulnerabilities coming from network dependencies are not novel, but its combination with vulnerabilities specific to machine learning could bring novel threats.

In comparison with ordinary IT systems, it is more important to securely develop machine learning systems in such a way not to incorporate vulnerabilities due to the difficulty of ex-post detection. When collaborating with other companies during the system development, or when using machine learning platforms or open-access databases, model developers should scrutinize the trustworthiness of partners or the validity of data sources.

C. Collaboration and strategy

In the practice of cyber security, system developers should collaborate with cyber security experts. Cyber security objectives cannot be accomplished solely by cyber security experts in the case of a machine learning system. Countermeasures such as modifying the model parameters affect the performance of the whole system.

Additionally, the planning and implementation of cyber security measures should be conducted in a strategically-designed way similarly to the case of the ordinary IT systems. [23] Though the security risks have not been captured yet

[22] Cyber security countermeasures have been proposed for some types of attacks. For example, for a perturbation attack against a machine learning model that performs image recognition, 'adversarial training' has been proposed in which adversarial examples are automatically generated and included in the training data. This enables simultaneous improvement in accuracy of the model and resistance to adversarial examples (Xie *et al*. [2020]).

[23] For example, Microsoft extended their 'threat model' approach for enumerating security risks in the ordinary IT systems to machine learning systems. They utilize it to guarantee the safety of products before launching new systems (https://docs.microsoft.com/en-us/security/engineering/threat-modeling-aiml).

comprehensively, the experts can perform better cyber security practices by developing a strategy referring to the taxonomy of the security risks.

D. Risk communication

For users, lawyers, and policy makers, as well as entities relating to machine learning services, understanding the security risks specific to machine learning is beneficial to facilitate risk communication among stakeholders. A machine learning system sometimes performs unexpectedly and could violate security characteristics. In this regard, as Kumar *et al*. (2019) points out, failure modes provide common concepts and languages as a basis for people to understand and discuss the security characteristics.

If machine learning systems would be implemented more deeply into social infrastructures, regulations on machine learning could become more desirable. For example, Kumar *et al*. (2018) and Calo *et al*. (2018) explore these policy options.

E. Concluding remarks

The machine learning approach is inductive in nature, and this calculation paradigm is different from that of conventional deductive approach. In the future, researchers are expected to deepen their studies on the vulnerabilities of machine learning systems and their associated security risks. Their taxonomy should be established reflecting such research to organize and stylize methodologies to mitigate the discovered security risks.

References

Amodei, Dario, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané, "Concrete Problems in AI Safety," arXiv: 1606.06565, 2016.

Athalye, Anish, Logan Engstrom, Andrew Ilyas, and Kevin Kwok, "Synthesizing Robust Adversarial Examples," *Proceedings of the 35th International Conference on Machine Learning*, 2018, PMLR, pp. 284-293.

Calo, Ryan, Ivan Evtimov, Earlence Fernandes, Tadayoshi Kohno, and David O'Hair, "Is Tricking a Robot Hacking?" University of Washington School of Law Research Paper 2018-05, 2018.

Carlini, Nicholas, and David Wagner, "Audio Adversarial Examples: Targeted Attacks on Speech-to-Text," arXiv: 1801.01944, 2018.

Ebrahimi, Javid, Daniel Lowd, and Dejing Dou, "On Adversarial Examples for Character-Level Neural Machine Translation," arXiv: 1806.09030, 2018.

Elsayed, F. Gamaleldin, Ian Goodfellow, and Jascha Sohl-Dickstein, "Adversarial Reprogramming of Neural Networks," arXiv: 1806.11146, 2018.

Fredrikson, Matt, Somesh Jha, and Thomas Ristenpart, "Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures," *Proceedings of ACM SIGSAC Conference on Computer and Communications Security (CCS) 2015*, Association for Computing Machinery, 2015, pp. 1322-1333.

Gao, Yansong, Bao Gia Doan, Zhi Zhang, Siqi Ma, Jiliang Zhang, Anmin Fu, Surya Nepal, and Hyoungshick Kim, "Backdoor Attacks and Countermeasures on Deep Learning: A Comprehensive Review," arXiv: 2007.10760, 2020.

Gilmer, Justin, Ryan P. Adams, Ian Goodfellow, David Andersen, and George E. Dahl, "Motivating the Rules of the Game for Adversarial Example Research," arXiv: 1807.06732, 2018.

Gu, Tianyu, Brendan Dolan-Gavitt, and Siddharth Garg, "BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain," arXiv: 1708.06733, 2019.

Hendrycks, Dan, and Thomas Dietterich, "Benchmarking Neural Network Robustness to Common Corruptions and Perturbations," arXiv: 1903.12261, 2019.

Jagielski, Matthew, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li, "Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning," arXiv: 1804.00308, 2018.

Kumar, Ram Shankar Siva, David O'Brien, Kendra Albert, and Jeffrey Snover, "Failure Modes in Machine Learning," arXiv: 1911.11034, 2019.

———, ———, ———, ———, and Salomé Viljoen, "Law and Adversarial Machine Learning," arXiv: 1810.10731, 2018.

Lee, Peter, "Learning from Tay's Introduction," Official Microsoft Blog, March 25 2016 (available at https://blogs.microsoft.com/blog/2016/03/25/learning-tays-introduction).

Leike, Jan, Miljan Martic, Victoria Krakovna, Pedro A. Ortega, Tom Everitt, Andrew Lefrancq, Laurent Orseau, and Shane Legg, "AI Safety Gridworlds," arXiv: 1711.09883, 2017.

Liu, Yingqi, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang, "Trojaning Attack on Neural Networks," Department of Computer Science Technical Reports, Paper 1781, Purdue University, 2017 (available at https://docs.lib.purdue.edu/cstech/1781).

Paschali, Magdalini, Sailesh Conjeti, Fernando Navarro, and Nassir Navab, "Generalizability vs. Robustness: Adversarial Examples for Medical Imaging," arXiv: 1804.00504, 2018.

Sharif, Mahmood, Sruti Bhagavatula, Lujo Bauer, and Michael K. Reiter, "Adversarial Generative Nets: Neural Network Attacks on State-of-the Art Face Recognition," arXiv: 1801.00349v1, 2017.

Shokri, Reza, Marco Stronati, Congzheng Song, and Vitaly Shmatikov, "Membership Inference Attacks Against Machine Learning Models," *Proceedings of the 2017 IEEE Symposium on Security and Privacy*, IEEE, 2017, pp. 3-18.

Tramèr, Florian, Fan Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart, "Stealing Machine Learning Models via Prediction APIs," *Proceedings of the 25th USENIX Security Symposium*, USENIX Association, 2016, pp. 601-618.

Xiao, Qixue, Kang Li, Deyue Zhang, and Weilin Xu, "Security Risks in Deep Learning Implementations," *Proceedings of 2018 IEEE Security and Privacy Workshops*, IEEE, 2018, pp. 123-128.

Xie, Cihang, Mingxing Tan, Boqing Gong, Jiang Wang, Alan Yuille, and Quoc V. Le, "Adversarial Examples Improve Image Recognition," arXiv:

1911.09665, 2020.