Humboldt-Universität zu Berlin
Mathematisch-Naturwissenschaftliche Fakultät
Institut für Informatik

Fraunhofer FOKUS

Risk assessment in Machine Learning security - a framework for risk measurement

Masterarbeit

zur Erlangung des akademischen Grades
Master of Science (M. Sc.)

eingereicht von: Jan Schröder
geboren am: 03.03.1996
geboren in: Lemgo

Gutachter/innen: Martin Schneider
Prof. Dr. Holger Schlingloff

eingereicht am: ..... verteidigt am: .....

Contents

Abstract

This thesis is Open Source and can be found on `https://github.com/EvilWatermelon/Risk-Measurement-Framework` together with the Risk-Measurement-Framework.

Acknowledgements

The following table explains and declares terms that are used in this thesis for standartization. The order of the table is organized by declaring a term before its relation to other terms.

Term	Description
International Organization for Standartization (ISO)	The ISO is an international statndard organization which declares international standards in all technical areas [3]. The ISO 27004 will be the basis for designing and implementing the Risk-Measurement-Framework.
Information Security Management System (ISMS)	The ISMS is a part of a management system that covers on the approach of a business risk approach the development, implementation, monitoring, review, maintenance, and improvement of information security [45]. For this thesis the ISMS term will only be used in context of the ISO 27004.
Model	A model in context of machine learning is a representation of a machine learning system that learned from the training sets [4].
Prediction	A predicition is an output of a machine learning model coming from an input e.g. data that the machine learning has never seen before [4].
Machine Learning	Machine learning is a research field and also describes a program or a system that trains from input data a predictive model. That predictive model makes predictions from never-before-seen data [4]. Machine learning models will be measured for security risks in this thesis.
Threat	A threat is a security violation when an event could cause harm [88]. Threats will be find in the threat models and the risk measurement.
Threat model	A threat model is a model to find security problems. This model should give a bigger picture away from the code [89]. A threat model will be used to declare different characeristics with their relation between attacks and an attacker.

Term	Description
Framework	A framework is a layered structure with a set of functions. It can specify programming interfaces, or offer programming tools [69]. In this thesis a framework will be designed, implemented and evaluated with a case study.
Vulnerability	A vulnerability is weakness of a program or system which could be exploited by a threat. [69]. The vulnerability term will be in this thesis used in context with the attacks and vulnerabilities in ML models.
Attacker	A person that exploits potential system vulnerabilities [69]. For this thesis an attacker stands in relation with its effort to attack an machine learning model.
Attacker's effort	The attacker's effort is a term that represent everything that an attacker do to attack the machine learning model. In this thesis the attacker's effort represent probability of occurrence, among other things.
Risk	Risk is the combination between the frequency of damage and the extent of damage. The damage is the difference between a planned and unplanned result [3]. The risk calculation will be used as the final result after the risk measurement.
Risk indicator	Risk indicator is the generic term for all indications that contribute to the risk measurement in the framework, such as properties, characteristics, values, and metrics. They will be used to measure risks and to represent the results as data, logs, or visual representations of the risk measurement.
Decision criteria	To describe the level of confidence in a given result, decision criteria pretend thresholds, targets, or patterns [2].
Measure	Variable which is assigned with a value that comes from a result of the measurement [2].
Analytical model	An algorithm that combines one or more derived measures with its associated decision criteria [2].

Term	Description
Measurement function	An algorithm to combine two or more measures [2].
Measurement method	Operations in a logical sequence [2].
Measurement	Gathering information about the IT security system by using a measurement method, function, analytical model and decision criteria [2].
Measurement result	One or more indicators with its interpretations which address an information need [2].
Information need	"Insight necessary to manage objectives, goals, risks and problems" [2].
Base measure	A measure that is defined by an attribute and the associate method for quantifying it [2].
Derived measure	"Measure defined as a function of two or more values of base measures" [2].
Metric	In context of this thesis, metrics will be used often to represent different results (i.e. accuracy) of the machine learning model.
High-level attributes	High-level attributes are subjective attributes that represent all characeristics of an attacker's effort [24]. This term will be used in context of a formal threat model.
Low-level attributes	Low-level attributes are objective attributes that represent all characeristics of the attack's data [24]. This term will be used in context of a formal threat model.

This thesis uses the terms decision criteria, measure, analytical model, measurement function, measurement method, measurement, information need, base measure, and derived measure in context with the ISO 27004 standard to design and implement the Risk-Measurement-Framework.

1. Introduction

Machine Learning (ML) is a constantly growing field and is essential for many innovative applications such as highly-automated and autonomous driving. Resulting from this, there is an increased need to maintain security. This thesis concentrates on risk measurement in context of common standards such as ISO 27004 which will be discussed in Section 2. This present thesis evaluates how to measure risks with the extent of damage and attacker's effort which should represent the probability of occurrence.

This thesis also explains and discusses the design of a conceptual framework and its implementation which is called Risk-Measurement-Framework (RMF). The RMF is designed by a conceptual framework based on risk indicators as a fundamental part upon approaches by Jakub Breier et al. [18] and Paul Schwerdtner et al. [84]. The core of the implementation of the RMF is the Adversarial-Robustness-Toolbox (ART) that is included as a Python library but also an open-source framework which will be explained in Section 2. Further, the risk indicators are evaluated during a measurement construct based on ISO 27004 to calculate the *Risk* and depict it in a risk matrix which structure bases from a standard of the European Telecommunications Standards Institute. The RMF only measures the values from backdoor attacks and evaluates them. This thesis do not propose any possible defense methods or suggestions for improvement of a attacked ML model.

Sections 1 and 2 are intended to clarify the goals and expectations of this thesis, explain terms, show necessary prior knowledge so that it is well defined where this thesis should go. Section 3 is one of the main parts of the thesis. The section discusses and describes the conceptual framework and gives the basis for the implementation of the technical framework explained in Section 4. Section 5 explains the case study that uses the framework and evaluates its results. Furthermore, Section 5 shows possible real-world scenarios that could use the RMF. In Section 6, a conclusion points out possible optimizations for the RMF, summarizes the results, and discusses them.

1.1. Motivation

The classic IT security is a large field and essential for every software application. In ML, security is also essential and needs more tools to find vulnerabilites and measure risks for the subsequent defense implementation. This thesis evaluates a conceptual and technical framework in the context of IT security standards. The aim is to improve security in ML, which could help researchers and companies to optimize their work. Due to the research for this present thesis, there were a lot of scientific papers that evaluate IT security management in the context of ISO 27005 but less with ISO 27004. Therefore, there is a need to put more focus on ISO 27004. So ML in relation to ISO 27004 is another motivating factor to extend the research in the context of security for ML and

ISO 27004. From the previously mentioned points, it should emerge that this thesis should show the possibility of using common standards for risk measurement in ML.

1.2. Research questions and hypotheses for this thesis

To understand the goal of this thesis the following hypotheses and research questions should show what this thesis is concentrating about.

Research questions

RQ1: How can the procedure and requirements from ISO 27004 be used for risk measurement of ML models?

RQ2: What are risk indicators to measure the extent of damage?

RQ3: What are risk indicators to measure the attacker's effort?

RQ4: Which measurement methods are suitable for measuring the extent of the damage and the attacker's effort?

RQ5: How are the results of a risk presented?

Hypotheses

In this thesis, the hypotheses represent assertions that have to be tested.

H1: There are several risk indicators to measure the extent of damage when attacking a ML model: **(a)** attack time **(b)** attack specificity **(c)** true positive (TP), true negative (TN), false positive (FP), false negative (FN).

H2: There are several risk indicators to measure the attacker's effort when attacking a ML model: **(a)** computational resources **(b)** attacker's goal **(c)** attacker's knowledge.

H3: It is possible to design and implement a risk measurement framework for ML models based on the generalized requirements of ISO 27004.

H4: The attacker's effort corresponds to the probability of occurrence when calculating the risk.

The first research question RQ1 is about the extent to which the ISO 27004 standard can be used for ML security. So the question should relate to what can be implemented and what cannot from the requirements. Thus, this research question is related to hypothesis H3. Research question RQ2 and RQ3 both refer to what should be measured. This refers to the hypotheses H1 and H2. The research question RQ4 should deal with the process after the research questions RQ2 and RQ3 on how the values of the measurement could be evaluated. The last research question RQ5 leads to the direction on how the results

of the risk measurement are presented that they are meaningful. For this purpose should hypothesis H4 discuss the relation between the attacker's effort with the probability of occurrence that is one value to calculate the $Risk$.

2. Related Work

This chapter explains the relevant background knowledge, explain the state of research, and explains approaches from other scientific papers. The first five subsections explain theoratical parts for this thesis in context to the concept and design of the RMF in section 3. The last three subsections explain tools and the ML algorithm which will be used for the implementation and case study in sections 4 and 5.

2.1. Security risks and risk assessment in context of Machine Learning

Security risks

Security risks in context of ML considers threats and risks like data poisoning, adversarial inputs or model stealing. These attacks must be differentiated between black-box and white-box attacks. Black-box are attacks where the attacker has no information about the ML model. In their work, Papernot et al. [71] explain a black-box attack strategy to misclassify a deep neural network (DNN) by generated adversarial examples. Papernot et al. assume that the attacker has no knowledge about the structure or parameters, and does not have access to any training set that this DNN uses. The attack strategy has no need to train an independent training dataset. The goal is to train a DNN with synthetic input data generated from the black-box adversary. Synthetic datasets are reflecting real datasets by preserving their relations between the variables and statistical properties [75]. The output data contain assigned labels from the targeted DNN and the adversary observes the output data. The input data are handwritten digits from the MNIST dataset and the output data is one of these digits. The aim of the attack is to get a minimal altered version of each input which is called an adversarial example. This should misclassify the input data by a minimal perturbation. This happens by generating the synthetic dataset which the adversary generates.
With white-box attacks have the attacker all information that he needs about the targeted ML model [92]. An example for white-box attacks is described by Prinz and Flexer [74], who present end-to-end adversarial attacks on classifications of music instruments. The training data are from the DCASE2019 Challenge [25]. In their work, Prinz and Flexer use four adversarial white-box attacks. Two attacks are untargeted and called Fast Gradient Sign Method (FGSM) [32] and Projected Gradient Descent on the negative loss function (PGDn) [58]. The other two targeted attacks are adaptions from the Carlini and Wagner (C&W) [20] method. The architecture is a convolutional neural network (CNN) [47] as Figure 1 shows.

PGDn is an attack on a negative loss function [57] which goal is to increase a sample of the negative loss interactively. FGSM is a method which goal is to increase the loss by moving a scaled step in a gradient direction while the adversarial pertubation is computed [32]. C&W's first method is about using gradient descent to minimise the loss to a specific class. The Multi-Scale C&W attack keeps pertubations small with the squared norm [20]. Prinz and Flexer [74] and Papernot et al. [71] show both ways to

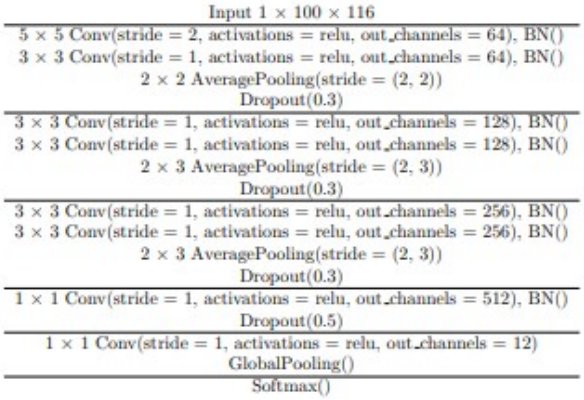

Figure 1: The neworks contains $5x5$ and $3x3$ convolutional layers with the ReLU activation, batch normalisation, and average-pooling layers. The CNN uses dropout for regularisation and the final layer is a $1x1$ convolutional layer adapted from [74].

attack ML models and in both works they used specific attacks for specific ML models. Adversarial inputs are data that are almost exactly the same inputs like the original data but classified incorrectly [58]. Duplicating a ML model via model extraction attacks is model stealing [39]. Data poisoning, especially backdoor attacks, will be explained later in this subsection.

Xiao et al. [100] evaluate the security risks in deep learning, for example identify the vulnerabilitesof TensorFlow. Xiao et al. use the framework sample applications along the frameworks. One statement of Xiao et al. is that the named frameworks TensorFlow, Caffe and Torch are implemented with many lines of code which make them vulnerable, for example heap overflow or integer overflow. To consider attack surfaces of deep learning applications Xiao et al. uses the MNIST handwriting digits [49]. The attack surfaces are divided in three angles. Malformed input is the first attack surfaces and describe input data for classification which is read out from files what the MNIST dataset does. Xiao et al. describe that input data from sensors like camera sensors is significantly reduced from vulnerabilites because the sensors are directly connected to the ML model. The second attack surface are training data where this thesis is specified on. The training data can be polluted or mislabled when they come from external sources. That pollution or mislabling is also known as data poisoning attacks [16]. These attacks may not base on software vulnerabilites but flaws in implementations make data poisoning easier. Xiao et al. describes an example where they observerd in image parsing procedures inconsistency in a framework and desktop applications like an image viewer. The incosistency makes a sneaky data pollution possible while the training process is monitored. The last attack surface are malformed ML models which are used models by developers that took them from other developers. Further, if ML models are designed and implemented from scratch and developers use them but does not have ML knowledge, attackers can manipulating them easier. This attack surface can also assigned to data poisoning attacks because attackers can attack external models if the developers are not defend

them from vulnerabilities. Beside of those attack surfaces, Xiao et al. describe different types of threats. The first threat are denial-of-service attacks which is the most common vulnerability and is caused by software bugs, infinite loops, or if memory can be exhausted. For example a bug in the numpy Python module is vulnerable and this module is the basis for TensorFlow. Another threat are evasion attacks that occur when the attacker use inputs that should classify as a certain label but instead it is missclassified as a different label. Evasion attacks are a possible attack if the ML model have software bugs (e.g. memory corruption bugs [67]). These bugs can be exploited when the classification can be overwited so the attacker can modify specific memory content or when the application control flow is hijacked to reorder the ML model execution. The last threat is system compromising with software bugs where an attacker hijack the control flow, leverage the software bug and compromise the host system.
Bagdasaryan et al. [9] evaluate more security risks in ML which are poisoning attacks in federated learning. Hayes and Ohrimenko [38] describe a contamination problem of data in multi-party models.

Poisoning Attacks

Poisoning attacks manipulate training data that are used to train ML models to misclassify the output data. Poisoning attacks can change the process while training but adversarial attacks can not. So poisoning attacks are able to manipulate the training datasets by poisoning features, flipping labels, manipulating the model configuration settings, or altering the model weights. The attacker has an impact on the training datasets or controls the training datasets directly. So the attacker wants to influence the ML model learning output [53].

Backdoor Attacks

Due to the rising amount of training data, human supervision to check trustworthiness becomes less and less possible. That exposes vulnerabilities in training datasets like backdoors. Backdoor attacks can cause far reaching consequences. The goal of backdoor attacks is to implement a pattern in an image which is called a backdoor trigger [34] to decrease the accuracy of inference. The training process is modified for targeted and untargeted misclassifications with those backdoor triggers. Then the labels are altered, the configuration settings are changed, or the model parameters are directly altered [53]. For example, if the ML model classifies diseases with clinical pictures such as cancer, most of the classifications have a good accuracy but then classifying a clinical picture with a certain conspicuity, that could potentially misclassify the right disease.

Risk assessment

Risk assessment in context of ML is derived from classic IT security risk assessment. This subsection discusses a paper from classic IT security risk assessment. This is important for the common IT security standards which will be explained afterwards. Sendi et al. [85] evaluates the taxonomy of risk assessment and at which point in IT security

management risk measurement takes place for this thesis and how it is carried out. Sendi et al. explain, that risk assessment is combined of risk analysis and risk evaluation [45]. In their paper, Sendi et al. evaluated 125 works published between 1995 and 2014. They developed categories for risk analysis such as appraisement perspectives, resource valuation and the last category is risk measurement. This category is the last step of risk assessment. To evaluate risks by measuring them, there are different properties which have an impact for risk measurement. Sendi et al. explain that the type of the attack, the dependency severity between resources and the type of defined permissions between resources are needed to measure risks. Risk measurement in their paper is differentiated between non-propagated and propagated. Non-propagated risk measurement stands in relation to the resource valuation category leading to the example of business driven risk assessment. Business driven is the view of business oriented goals and processes. And non-propagated risk measurement means that a model in which the risks are measured without the impact from other resources. For example, if the risks are measured business driven, the parameters such as business process are seen without the impact from other business processes. Propagated risk measurement concentrates on the attack impact and its propagation on other resources [42], [46]. The risk measurement is measuring the propagated risks as a dependency graph. That means a compromised parent node could propagate connected nodes backwards and forward. Backward impact means the impact propagation on all nodes that have a dependency with the compromised node and forward impact is the propagation from the compromised node to all its dependent nodes. In context to ML the propagated risk measurement is important, for example because a manipulated training and testing dataset from an external source could lead to a misclassification while training and testing in the ML model that represents the parent node.

2.2. Relevant standards for risk measurement

As a basis, this thesis uses the requirements of ISO 27004. ISO 27004 is an international security standard from the ISO 27000 family which guides continuous basis evaluation methods. This standard stands in no context with ML but will be derived in this thesis to design and implement the RMF in section 3 and 4.

ISO 27000 family

In their book, Kersten et al. [45] explain and discuss the management of the information security based on the ISO 27000 standard. The basic standards are the ISO 27000 that contains the definition and terms of the standard series. ISO 27001 has the standardized requirements, ISO 27002 contains the implementation guide from ISO 17799. ISO 27003 specifies the implementation of an IT security system. ISO 27004 shows the requirements for measurement which contains the metrics and key figure systems. ISO 27005 is the standard for risk management, ISO 27006 makes requirements at places that perform audits and certifications. ISO 27007 contains security system audits, ISO 27008 makes requirements on technical audits and ISO 27010 shows how to do an exchange of security

informations. There are ten more ISO 27k standards but these are for special sections and none of them contain ML itself or in context of security. Figure 2 shows the relation between the standards without special sections.

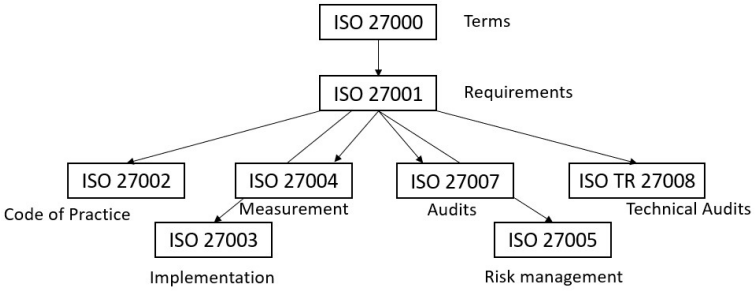

Figure 2: Overview of the ISO 27000 without special sections adapted from [45].

ISO standard for risk measurement

Kersten et al. [45] explain if a security system wants a certification then ISO 27001 must be fulfilled. The other related standards shown in figure 2 are optional and are not bound to get the certification. For the RMF, ISO 27004 is the standard to measure risks and the other ISO 27000 standards are not considered further.

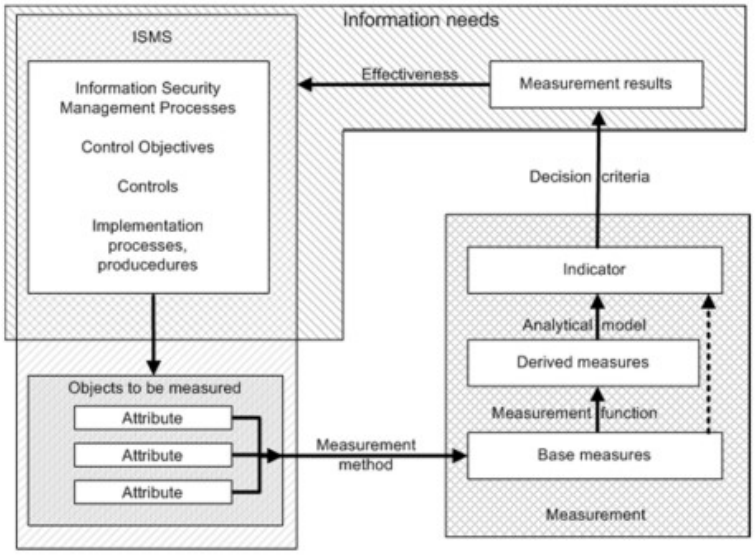

Figure 3: The information security measurement model adapted from [93]

This ISO can be related with ISO 27001 or used as a standalone standard. As a requirement in ISO 27001, the effectiveness of an IT security system must be measured [10]. The ISO 27004 standard specifies, what to be measured, when the measurement is needed, and types of measurement [54]. Barabanov et al. [10] and Tarnes [93] describe in

their works the different properties of ISO 27004 for risk measurement. Tarnes shows the information security measurement model which is shown in Figure 3. The measurement model in Figure 3 explains the relevant properties and its conversion to indicators that give a basis for decisions. The relevant properties show the needed information to the measurement objects.
An object of measurement contains different attributes that are selected for the measurement method, resulting in the base measure. A measurement method is an algorithm to quantify attributes. The derived measure is a summary of at least two base measures. To get this derived measure from the base measures, a measurement function combines them based on different scales such as qualitative assessment results and percentages. An analytical model derives an evaluation or estimate of specified attributes to get the indicator measure. The analytical model applies a base and a derived measure. Based on defined decision criteria, the indicators can be interpreted to develop measurement results. ISO 27004 describe decision criteria as needs for further investigation or actions to define the level of confidence in the measurement results [2].

ISO 27004 specifies the requirements how to develop measures and the measurement in the following list [2]:

(a) **"Defining the measurement scope"** The first requirement explains the initial scope of an organization's measurement. It is based on different elements depending on capabilities and resources of the organization. These are specific controls and their protected information assets and information security activities. The management must prioritize all of them. Furthermore, the internal and external stakeholders should be identified and should participate on the measurement scope. It is possible that the organization sets a limit on the number of measurement results. This should ensure that the decision-makers can improve the ISMS based on the measurement results in a given time interval. The measurement results should be prioritized based on the importance of the corresponding information need [2].

(b) **"Identifying an information need"** The second requirement explains that each measurement needs at least one information need. An information need is identified in four activities. At first, the ISMS and its processes must be examined. Then the identified information need must be prioritized based on criteria, such as risk treatment, the organization's capabilities and resources, the interest of stakeholders, and the information security policy. The third activity bases on the list of prioritized information needs where a subset of information is required to be addressed on the measurement activity. The last activity is the documentation and communication to the stakeholder of the selected information need.
Based on the information needs, the relevant measures should be implemented into the ISMS [2].

(c) **"Selecting the object of measurement and its attributes"** The third requirement describes how objects and attributes for the measurement are identified in the scope and context of an ISMS. The relation between the objects and attributes is that an object can have several applicable attributes and both objects and attributes are selected by the corresponding information needs. Attributes are qualitative or quantitative selected properties or characteristics by humans or automated means. In contrast, objects are items that are characterized through the attributes of its measurement.
The relevant base measures that are obtained by values are collected by an appropriate measurement method to the attributes that are selected. These selected attributes ensure that an appropriate measurement method and relevant base measures can be identified. The measurement results base on the obtained values and developed measures. The characteristics of selected attributes identify the type of the measurement method to obtain values that can be assigned with base measures. All of the chosen objects and attributes need to be documented. Objects and attributes that are described by data should be used as values that are assigned to the base measures. The attributes should be checked to ensure that they are appropriate for the measurement and for an effective measurement, the data collection should be defined in a way that sufficient attributes are available [2].

(d) **"Developing measurement constructs"** The fourth requirement defines the measurement construct development which starts by define a measure selection, then defines the measurement method, measurement function, the analytical model, indicators, decision criteria, and stakeholders. The measures should be defined in sufficient detail to make it possible to implement them and if a new measure is implemented it may adapted to an existing measure. Selected measures should represent the information need's priority. Example criteria are, facilitation for data collection, facilitation for interpretation, and measures to calculate costs of analysing, and collecting the data. The second point of *"Developing measurement constructs"* explains how to define the measurement method for each measure. That measurement method will be used to quantify the measurement object by transforming the attributes into the value that is assigned to the base measure. Further, the measurement method can be subjective or objective. Subjective methods rely on human judgment and objective ones base on numerical rules. In the measurement method, the attributes are quantified as values. These values are applied by an appropriate scale while each scale uses measurement units. Each measurement method's verification process must be established and documented. Furthermore, the precision of the measurement method and its deviation or variance should be recorded. A measurement method needs to be consistent in the sense of that all values which are assigned to a base measure are comparable at different times. These values should be also comparable to derived measures and indicators. The next part is about the measurement functions (e.g. calculations). The measurement functions may combine different techniques, for example averaging all

values that are assigend to a base measure. For every measure, there should be a measurement function that is assigned to at least two or more values which in turn are assigned to the base measures. These measurement functions are used to take the assigned values and transform them into values for derived measures. The analytical model is defined for each indicator by transforming values that are defined to a base or derived measure. The analytical model creates outputs that are relevant for all stakeholders. These values should be transformed into a value that is assigned to an indicator. Indicators are assigend values that are assigned to aggregated values which in turn are assigned to derived measures. These values are interpreted based on the decision criteria. The decision criteria are defined and should be documented based on information security objectives. Decision criteria are based on historical data, plans, and heuristics or calculated as statistical control or confidential limits. Lastly, stakeholders from base or derived measures are identified. Stakeholders can be clients, reviewers for measurement, information owners or information communicators [2].

(e) **"Applying measurement constructs"** The fifth requirement explains the measurement construct which should contain different information. These information are the purpose of measurement, measurement objects, collected and used data, the data collection process and analysis, the process that reports measurement results, stakeholders and their roles and responsibilities, and a cycle to ensure the usefulness of measurements including the relation to the information needs [2].

(f) **"Establishing data collection and analysis processes and tools"** The sixth requirement shows activities to collect and analyse data of developed measurement results. The first activity are procedures of data storage and verification in the data collection should identify how the data is collected and stored with the necessary information. The collected data should come from designated measurement methods. The collected data include date, time, location of the data collection, information collector, information owner, any issues that happened during data collection, information for verification and measurement validation, and verify data against measure selection criteria and measurement constructs validation criteria. This should be done by using measurement methods, functions and analytical models.
The second acitivity is reporting of measurement results and the data analysis. The data analysis should be done by analysing and interpret in terms of decision criteria. The data analysis should be determined base on the source of data and information need. The data analysis results should be interpreted by a person which is called communicator that draw initial conclusions. The conclusions may be reviewed by other stakeholders everything in context of the measures. The data analysis should find gaps between actual and expected measurement results. The gaps should show needs to improve the ISMS, including the scope, policies, objectives, processes,

procedures, and controls [2].

(g) **"Establishing measurement implementation approach and documentation"** The last requirement shows the amount of information which should at least be part of an implemententation plan [2]:

a) Information Security Measurement Programme in the organization

b) Measurement specifiations:

i. Organization's generic measurement, and

ii. Organization's individual measurement construct

iii. Range and procedures for data collection and analysis definition

c) A plan as a calendar for measurement activities

d) Created records through measurement activities with collected and analysis data records

e) Formats for measurment results that are reported to stakeholders (described in ISO 27005)

2.3. The threat model for attacker characteristics

This subsection explains a threat model without a reference to ML and which is used in the RMF. In their paper, Doynikova et al. [24] describe a formal threat model with its attributes. This threat model will be used for the RMF. Doynikova et al. suggest to split the theat model into high-level and low-level attributes. High-level attributes are subjective and can not obtained from monitoring the system under analysis. The monitoring system is described as features from network traffic and event logs. These logs and the traffic are described later in this subsection. The gathered data are divided in four groups. The first group includes characteristics such as skills, motivation and intention. The second group characterizes the attacker's capabilities and show the characteristics as used resources. The third group incorporates the attacker in relation with the attacked system. This group includes the attacker's location, the privileges, his goals, the access and the attacker's knowledge. The attacker's knowledge comes from the system where the objects are accessed before, access and privilege type and the detected activity. The last group relates the attacker with the attack and the steps that are included to execute the attack.
The low-level attributes can be used directly from the raw data during monitoring the system. Doynikova et al. consider these attributes as objective. The attributes are classified into event logs, network traffic, namely, and their source. The event log and network traffic is classified by origin, target, content, and temporal characteristics [27] which are explained next. The attackers goal, target of the attack or a normal action is monitored by the target characteristics. An attack is specified by content characteristics. Temporal characteristics contain time characteristics of the attack on a specific time

interval and incorporate frequency. The observable attack characteristics incorporate observables from the attack that are not in the other four characteristics for example from the event logs.
Based on the low-level attributes, the high-level properties can be calculated by mapping the low-level to the high-level attributes like the attackers skills, resources and motivation. This mapping is a challenge for this thesis and the chraracteristics of both the high- and low-level attributes have to be derived for the RMF. Sections 3 and 4 map these attributes in the RMF.

2.4. Machine learning metrics

In their book, Nguyen and Zeigermann [63] describe that ML metrics are different calculations to evaluate a ML model. These metrics are linked to the loss function. A loss function serves as an optimization process and is used by the optimization algorithm to adapt while the next iteration iterates through the parameters of the model. The more a loss function is decreased, the better is the result. Further, the score is a value that is calculated while testing right after the training. The score and loss functions are summarized as metrics. In regard to these definitions, the terms score metric and loss metric will be used for the remainder of this chapter. The first score metric is the accuracy. The accuracy relates the number of data examples with true predicted labels to the number of all examined data examples.

$$accuracy = \frac{n(correctly\_predicted)}{n(all)}$$

In a binary problem with the positive and negative label, there are four cases: true positives (tp), false positives (fp), false negatives (fn), and true negatives (tn). These labels can be summarized in a $2x2$ confusion matrix.

$$\begin{pmatrix} tp & fp \\ tp & tn \end{pmatrix}$$

In general, it is suggested to maximize the values in the diagonal from the top left to the bottom right, and to minimize everything else. The values in the diagonal show the number of test examples that the ML model predicted correctly. All other values are wrong predictions. Further, to get the rates of the confusion matrix, the matrix values have to be divided by the number of examples per category.
The last metric is the precision-recall:

$$Precision = \frac{tp}{(tp+fp)}$$

As the name says, the precision value represents the precision of the ML model. It calculates the tp rate.

$$Recall = \frac{tp}{(tp+fn)}$$

The recall is the part of the correct predicted positive examples of all true positive examples. This value can be interpreted as the ML model efficiency. A third score for the balance between the precision and recall is the F1-Score.

$$\frac{2*precision*recall}{(precision+recall)}$$

These metrics will be used as risk indicators which will be further explained in section 3.

2.5. Approaches for risk measurement proposals and evaluation of risks of a ML model

This thesis is divided into two approaches. Breier et al. [18] propose in their paper different proposals to measure risks with different aspects. These aspects are used in this thesis as proposals for for the risk indicators and will be extended in subsection 3.4. These different proposals are attack specificity, attack time, attacker's knowledge, and attacker's goal. Attack time is split in training time and deployment time. Training time is the attack time when the model gets manipulated while it trains. Deployment time is the attack time when the hacker attacks a ML model after its release. Attacker's knowledge is the amount of information the hacker has available. Attacker's specificity is the amount an attacker needs to manipulate the output of a ML model. Attacker's goal is a classification about what the attacker try to achieve with an attack. These four proposals may serve as a basis for further risk indicators for risk measurement. Since these suggestions overlap with the characteristics from the threat model of Doynikova et al. these suggestions can be raised from the low-level and high-level properties. The work of Breier et al. do not concentrate on the attack, which is a research gap this thesis is trying to fill.
Paul Schwerdtner et al. [84] is the second approach of this thesis. Schwerdtner et al. show a technical framework to evaluate the risks for ML models. Schwerdtner et al. give an evaluation whether it is secure to deploy a ML model or not. The ML model in Schwerdtner et al. must be a fully developed ML model that is trained and tested. Schwerdtner et al. concentrate on input data when the ML model has finished training and testing. The technical framework can test ML models under specific conditions in a scenario but can not find or measure risks while the training process. At this point the RMF would then find use.
This third approach is for poisoning attacks. Biggio et al. [16] describe an investigation on poisoning attacks against SVMs. Attacks against ML models are classified in causative and exploratory attacks [14]. Causative attacks are manipulations against training data and exploratory attacks are exploitations against the classifier. Poisoning attacks are refered to causative attacks. These attacks are important when the attacker have no direct access to the training database and provides his own training data from a web-based repository. Biggio et al. explain that the attacker's goal on a data poisoning attack is to find a point where the SVM's accuracy can be maximally decreased. Further, most learning assume that training datasets come from natural or well-behaved distributions.

In their work, Biggio et al. assumes a worst case of the attacker's capabilities. The attacker also knows the ML model and can get the data from from a underlying data distribution platform. That is the security analysis methology [13]. Biggio et al. shows a method where the attacker construct a data point which decreases the SVM's classification accuracy. This method base on the on the SVM's optimal solution of the training problem. The training problem is formulated as the quadratic programming problem and for further explanation, please see [70].
An attacker can attack this SVM by inserting special attack points. Biggio et al. shows such a way assuming that the optimal solution of the SVM's training problem is retained. The SVM must be trained and the training points are as margin, and error support vectors, and reserve points referred. The attack is initialized by an attack vector which clones and arbitrary point from the attacked class and flipping the label [103] from the point. For this thesis this work is a very generalized approach. There is no explanation of the type of poisoning attack and it does not show what affect this attack have on real-world scenarios. This approach goes more into the implementation and evaluation parts of the thesis and should show a first method how to attack a SVM.

2.6. Adversarial-Robustness-Toolbox

For this thesis the technical framework Adversarial-Robustness-Toolbox [66] is a main component. Nicolae et al. [65] evaluate in their work the technical framework ART. ART is a Python library that supports several ML frameworks for example TensorFlow and PyTorch to increase the defense of ML models. It is designed for developers who want to secure ML models. ART support 39 attacks and 29 defense functions. The attacks are evasion, extraction, inference, and poisoning attacks. The defences are detector, postprocessor, preprocessor, trainer, and transformer defences. This thesis only focuses on the attack functions for poisoning attacks. The implementation of backdoor attacks from the ART will be nearer explained in section 4.

2.7. Keras

In this thesis Keras [33] is the main technical framework to implement the case study. Keras is a so called high-level neural network Application Programming Interface (API) which is implemented in Python. High-level frameworks are used to increase the flexibility to design a ML model and simplify the implementation [60]. The second level behind Keras are low-level frameworks such as TensorFlow, PyTorch and MxNet which are called Backend-Engine in Keras. Keras converts its implementation to TensorFlow which runs the ML model. The hierarchical structure behind Keras is shown in Figure 4 with possible frameworks that Keras can use. Keras can be used without any changes in the Backend-Engine and be executed on the CPU and GPU [26].

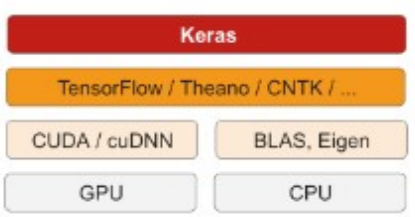

Figure 4: Hierarchical structure of Keras adapted from [26]

TensorFlow

TensorFlow [5] is the low-level framework that is used behind Keras [60] for this thesis. It executes with dataflow graphs for computation representation, shared state, and with operations to mutate the state. Dataflow makes it possible to communicate between subcomputations and executing parallel independet computation. In TensorFLow all data is modelled as tensors which are n-dimensional arrays where each element have a small number of primitive datatypes. Furthermore, TensorFlow uses operations where one operation takes $m \geq 0$ inputs and computates $n \geq 0$ outputs.

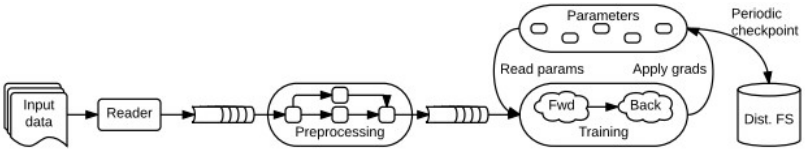

Figure 5: TensorFlow dataflow graph that shows a training pipeline with subgraphs that take input data, preprocessing, training, and checkpointing state - adapted from [5]

2.8. Neural network

A neural network (NN) [56], [105] is a ML model which basis is a hierarchical neuron organization where the neurons are connected. In the neurons the computation for the output is executed. The first layer are input data which are called tensors. Tensors are vectors in a n-dimensional matrix. The input data can be various types but only assigned as numeric data. The following example shows a mxn shape which represents a two dimensional tensor (vector a_{11} to a_{m1} is one training sample):

$$A_{m \times n} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

The first hidden layer gets inputs from the input layer which receives the input data. The neurons in the hidden layer have a activation function which calculates the input

data multiplied with weights. Every ML model can have a stack of hidden layers one after another.

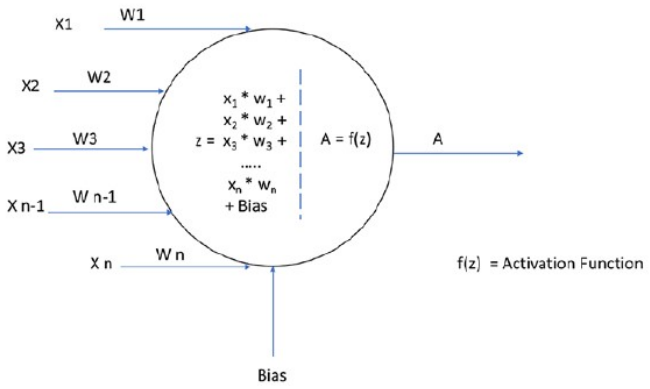

Figure 6: A single neuron adapted from [60].

The most common activation functions [59] are the rectified linear unit (ReLU) function $Relu(z) = max(0, z)$ and the sigmoid function $\sigma(z) = \frac{1}{1+e^{-z}}$ which are shown in Figure 7 and 8.

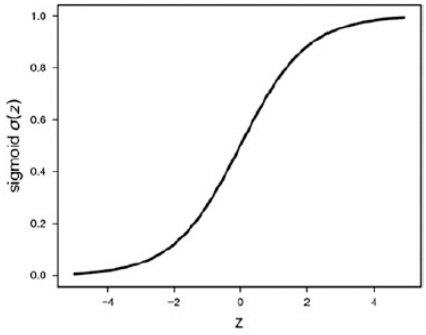

Figure 7: Sigmoid function adapted from [60]

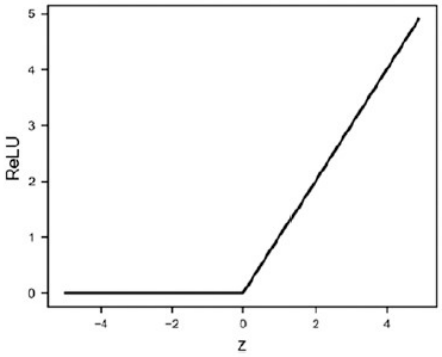

Figure 8: ReLu function adapted from [60]

After the hidden layers the NN has an output layer which shows the output as the name says. The next part of a NN is the loss function metric to measure the target loss. Popular loss functions [98] are the mean squared error $\sum_{i=1}^{D}(x_i - y_i)^2$ or the mean absolute error $\sum_{i=1}^{D}|x_i - y_i|$.
The main part of the ML model training is the optimizer which optimizes the ML model with an algorithm that is called backpropagation. Backpropagation executes after the ML model has started with the activation function and random weights in the defined structure explained earlier in this section. After this structure the loss function takes the ouput data to tell the NN how well it has predicted the input data based on the

used weights. To update the weights for each neuron to reduce the loss, the optimizer algorithm uses a chain rule in calculus, derivates, and partial derivates. An example of an optimizer is stochastic gradient descent $Weights = Weights - learningrate * Loss$. The learning rate must be defined as a parameter in the NN architecture.

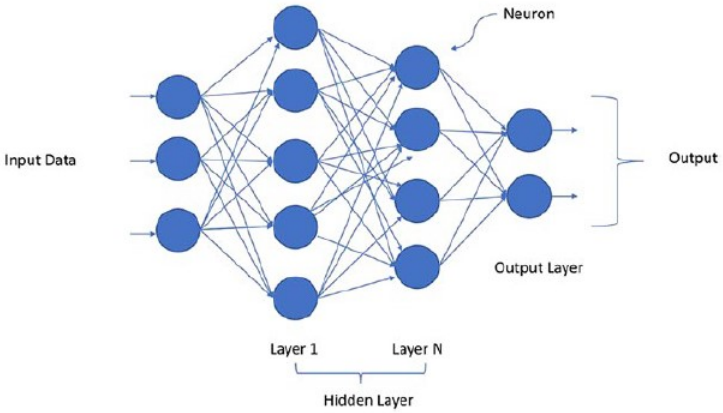

Figure 9: NN example adapted from [60].

3. Risk Measurement Framework concept and design

In contrast to Schwerdtner et al. [84], the framework of this thesis concentrates on training, especially risk measurement before and during training of the ML model. This section discusses and explains the design of the RMF. The RMF is a technical framework based on a concept to measure risks of backdoor attacks [8] by measuring the attacker's effort and extent of damage of an attack.
Referencing on this thesis approach from Biggio et al. [16] a security analysis for ML is that the attacker knows the ML model and can use the data from the data distribution platform. It is assumed for the RMF that the attacker knows the training data. This is an unrealistic assumption, but in real-world scenarios the attacker could use a surrogate training set instead, from the same data distribution platform which the developers use [13]. The following subsection goes to the research question RQ1 - How can the procedure and requirements from ISO 27004 be used for risk measurement of ML models?

3.1. Using the standards for the risk measurement

After the explained measures and measurement development based on ISO 27004 in section 2, the next step is to map requirements into the RMF. This discussion what parts of them can be fulfilled and which parts can not fulfilled before using the RMF.

"Defining the measurement scope" is the first requirement about the organization's capabilites and resources which define the initial scope. It starts by decisions of the management and can not be fulfilled by the RMF because that is an individual process specific for an organization and stands not in relation with the risk measurement in the RMF. The part defining stakeholder can not be fulfilled but in "Developing measurement constructs" it will be further discussed how to identify them.

"Identifying an information need" is as the name says about the identification of an information need. The first activity of identifying the processes and examination of the ISMS can not be fulfilled because the RMF stand in no relation with an ISMS directly. The information need prioritization criteria such as risk treatment can be fulfilled by the general riks measurement because all results are shown as transparent as possible. The organization's capabilities and resources criteria is individual for the organization and can not be measured. The interest of stakeholders are individual and must be defined before using the RMF. The third activity can be fulfilled by showing all results of the risk measurement which appendix C shows. The last acitivity is a process about documenting and communicating with the stakeholder which can be fulfilled after using the RMF because every risk measurement step is documented.

"Selecting the object of measurement and its attributes" describe that objects

and attributes are identified in the scope and context of an ISMS. The identified objects and attributes are related to the risk indicators which are used to measure risks and calculate the final risk. As Figure 10 shows, the risk indicators are represented by objects and attributes in the ISO 27004 [2] standard. In order to assign the terms of objects, attributes, and base measures to the RMF, all risk indicators are assigned to these standard terms. When this allocation takes places, this is always mentioned.

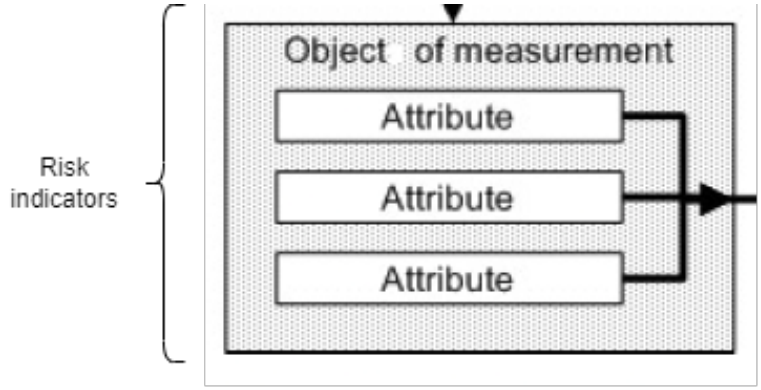

Figure 10: Relation between the objects, attributes and the risk indicators adapted from [2].

The terms of the standard thus enable a better classification and relationship of the terms assigned to the risk indicators. In order of this requirement, attributes identify the type of measurement methods to obtain values which are assigned to the base measures. To fulfill the relation between the measurement methods that are selected through the attributes, there is a need to relate the attributes with a measurement method. For more detailed explanations of the measurement methods the following subsections go into the individual points of the concept for the RMF.

"Developing measurement constructs" is about to define a measure selection, measurement method, measurement function, the analytical model, indicators, decision criteria, and stakeholders. Starting by identify a measure selection the following example criteria from ISO 27004 should help: facilitation for data collection, facilitation for interpretation, measures to calculate costs of analysing, and collecting the data. The data collection can be done through the measurement methods. All of the collected data should be used for interpretation, for the attack, and the attacker's effort. Specificlly on poisoning attacks the training data must be analyzed and any information can be used for the measurement process. It is complicated to find poisoned data because the modifications are small in images which can be harder to get these information. The success depends on the patch size and the attacks on an image are highly specified [83]. Therefore it is clearer to define the risk indicators as the measure selection because it measures certain values that are common for a particular attack.
The measurement method is used to quantify the measurement object by transforming

the attributes into the value that is assigned to the base measure. Measurement methods can be subjective or objective. The objective measurement methods measure based on the data that are given from an attack. The subjective measurement methods are based on human judgment as, for example, with the threat model of Doynikova et al. [24].

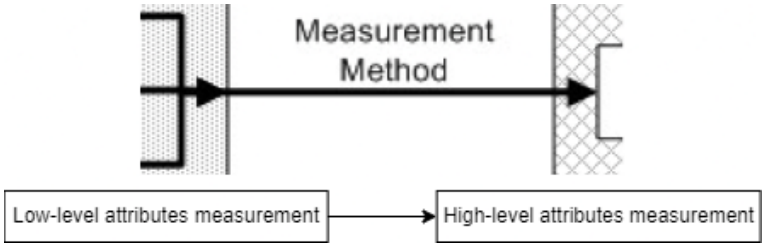

Figure 11: The measurement method (right) send the collected data to the measurement method left (left). Both measurement method are executed at the second step of the measurement from ISO 27004 - adapted from [2].

The outputs of the measurement methods need to be assigned to the base measures. Every base measure is an attribute from the objects but with an assigned value in the RMF. These base measures are used as the input for the measurement functions which are calculations to transform them into derived measures or can be directly handed over to the analytical model. The measurement functions are calculations to transform the base measures into derived measured. In context to the risk measurement of the RMF this can be fulfilled for values which are not calculated already for example, the accuracy is a risk indicator that is already a total value [63] and do not need a further calculation with other results from the measurement method's output.
After explaining the derived measure the next step in the measurement is the analytical model. For each indicator there should a analytical model by transforming values that are defined to a base or derived measure. Indicators are assigend values to aggregated values which in turn are assigned to derived measures and/or base measures. The analytical model creates outputs that are relevant for all stakeholders. However, this connection to the stakeholders is not considered further in this thesis. The values that are assigned to indicators and how they are presented describe "Establishing data collection and analysis processes and tools".
If the indicators are evaluated the decision criteria are the penultimate step before the measurement results. Decision criteria are based on historical data, plans, and heuristics or calculated as statistical control or confidential limits. That process can not be fulfilled because the basis of the decision criteria is a part which should be present before executing the RMF on a ML model.
The last requirement is defining stakeholders. Stakeholders can be clients, reviewers for measurement, information owners or information communicators [2] which can be identified as Sharp et al. [87] explain in their work.

Figure 12 shows the process (of this requirement) after the measurment methods are finished and assigned their output to the base measures.

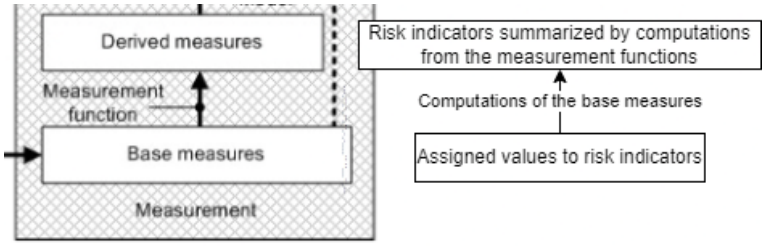

Figure 12: After the measurement methods are finished, the outputs are assigned to base measures and named as the risk indicators. A few risk indicators are calculated to derived measures - adapted from [2].

"Applying measurement constructs" is the requirement that explains which information the measurement construct should contain. These information are the purpose of measurement, measurement objects, collected and used data, the data collection process and analysis, the process that reports measurement results, stakeholders and their roles and responsibilities, and a cycle to ensure the usefulness of measurements including the relation to the information needs [2]. In context to the RMF the stakeholders and their roles and responsibilities, and the cycle to ensure the usefulness of measurements including the relation to the information needs can not be fulfilled. These information are not gathered from the RMF and it is not the goal of the RMF to collect these information.

"Establishing data collection and analysis processes and tools" is the process of collecting and analysing data to identify how the data is collected and stored with its necessary information in the developed measurement results based on two activities. This process can be fulfilled in all individual processes of the risk measurement in the RMF. The first acitivity is how to store the collected data. The necessary information are date, time, location of the data collection, information collector, information owner, any issues that happened during data collection, information for verification and measurement validation, and verify data against measure selection criteria and measurement constructs validation criteria.
The second acitivity can not be fulfilled by the RMF because it requires a communicator and stakeholders which analyse and interpret the data by human judgment.

"Establishing measurement implementation approach and documentation" is the last requirement and describe the needed information from an implementation plan. This plan is not the basis of the RMF because the process depends on organization's specifications and plans as a calendar which this thesis not using to design and implement the RMF. Instead of this implementation plan, subsection 3.13 show the final structure after which the RMF is implemented.

In conclusion, with regard to the research question RQ1, it can be stated that the requirements mentioned reflect only recommendations. That makes it possible to fulfill the requirements for security improvements in ML and confirms the hypothesis H3 regarding to the requirements and procedures of ISO 27004. The following sections 3.2, 3.3, 3.6, 3.7, and 3.8 explain the concept of the measurement methods in the RMF.

3.2. Characteristics of backdoor attacks

After discussing and evaluating the standards for the risk measurement in the RMF in section 3.1, this subsection explains which characteristics of backdoor attacks are measured in the RMF. Biggio et al. [16] explain that poisoning attacks and therefore also backdoor attacks are causative attacks which means manipulations against training data is the focus of them. Further, Xiao et al. [100] describe that training data can be polluted or mislabled when they come from external sources. Xiao et al. explain that poisoning attacks are not based on software vulnerabilities, which means that software bugs are not the execution point of backdoor attacks when implementing them into the RMF. This means the RMF measures conspicuities in the training data and measures the computational resources. Furthermore, the RMF measures before and after an attack to compare the differences between the original and manipulated ML model.

3.3. Types of backdoor attacks

The following backdoor attacks should be used to represent possible attacks on a ML model. Further, this subsection should show the main principles of the backdoor attacks that are used in the RMF.

Backdoor attack concepts

Pattern Backdoor Attack, *Clean Label Backdoor Attack*, and *Hidden Trigger Backdoor Attack* are the three backdoor attacks for this thesis.

Pattern Backdoor Attack Gu et al. [34] explain *Pattern Backdoor Attack* which goal is to change original labels to a target label on outsourced ML models. This happens by attacking a random small selection of the training set and apply a backdoor trigger into the input data. To be more precise, a backdoor attack works by adding a trigger in form of a pattern into some images in the training set, depending on the attack specificity (targeted or untargeted). Targeted means to poison specific images of a label that should trigger the backdoor if a pattern is on the images. Untargeted is an attack specificity through which images are random from a random label.
After a successful attack it is difficult to detect backdoor attacks because the ML model's performance does not change noticeably in relation to the original performance [44]. Further, backdoor attacks are powerful, because they take control over images that should be misclassified while the ML model is in test time [94]. In their work, Gu et al. show

different backdoor attacks and perform a case study with a traffic sign detection attack on a neural network. The evaluated backdoors are a single pixel backdoor and a pattern backdoor. The single pixel backdoor increases the brightness of a pixel and the pattern backdoor adds a pattern of bright pixels in an image which Figure 13 shows.

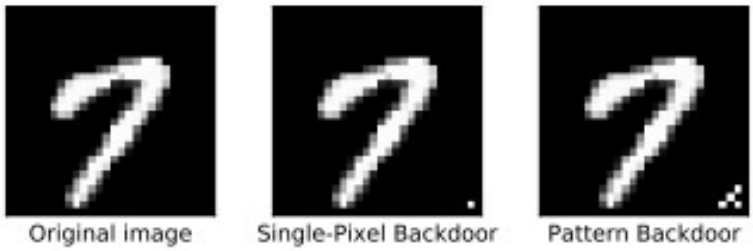

Figure 13: Backdoors in relation to the original image adapted from [34].

The implemented attacks from Gu et al. are a Single Target attack and an All-to-All attack where the training data is poisoned [40]. Single Target attacks use the single pixel backdoor by mapping a label from a digit (that is an image of the MNIST [50] training set) i as a digit j on backdoored inputs. Figure 14 represents the classification error of the color-coded values where the row is i and the column j.

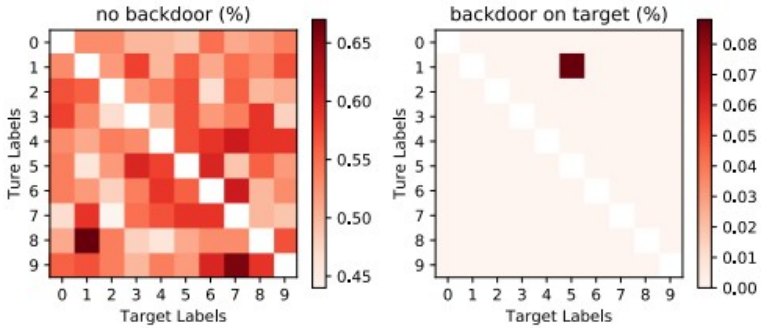

Figure 14: The left side shows the classification error (%) of clean images with a Single Target attack for every instance. The right side shows the same but with backdoored images - adapted from [34].

The attack strategy is a random pick of images from the training data and implements a poisoned version back to the training set. An All-to-All attack changes a digit label i to $i + 1$. After testing the All-to-All attack, the original ML model has an error rate of 0.03% while the ML model with the backdoored image has an average error of 0.56%. The case study is a traffic sign detection attack where the label of a stop sign is changed to a label of a speed limit sign. The backdoor of the image is either a yellow square, bomb image, or a sunflower image as the size of a post-it note on the stop sign. These backdoors are found at the bottom of the stop sign.

The setup for the case study of Gu et al. bases on the Faster-RCNN [76]. The Faster-RCNN takes an image as input data and the output data is a proposal as a set of rectangular objects where every rectangle has an objectness score.

Figure 15: Stop sign as the clean version and with the three backdoors adapted from [34].

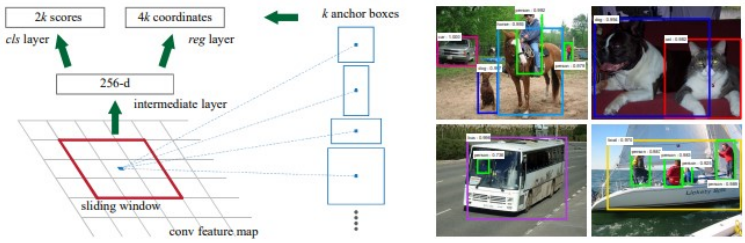

Figure 16: The left side shows a region proposal network which represents the Faster-RCNN network and the right side shows an example of the detection adapted from [76].

The attack is a Single Target attack where the label of the stop sign image changes to a speed-limit label, and a random target attack where the stop sign is randomly changed to an incorrect label. The results show a successful pass of the validation tests and 90% of the stop signs are missclassified as speed-limit signs. Gu et al. tested their ML model in a real-world attack by pasting a sticker on a stop sign near their office building. The ML model classified the stop sign with a 95% confidence as a speed-limit sign. The ML model's accuracy is decreased on backdoors to 1.3% which makes a misclassification to more than 98%. This attack is now transferred to the RMF for risk measurement to check how much the accuracy is reduced. The more the accuracy for the classification of the poisoned images is decreased, the higher is the possible extent of damage which increases the risk.

Clean Label Backdoor Attack In their work, Turner et al. [94] explain *Clean Label Backdoor Attack* attacks in direct comparison to Gu et al. *Pattern Backdoor Attack*. Turner et al. show an approach for executing backdoor attacks by utilizing adversarial examples and Generative Adversarial Network (GAN)-generated data. The approach of Turner et al. is analyzing the effectiveness of the attack of Gu et al. while a simple technique is applied for data filtering. Data filtering is a technique to find poisoned data by comparing the used training data with trusted training data. Turner et al. discovered that the poisoned inputs are outliers and are clearly wrong from the human inspection side. The attack would be ineffective if it would rely solely on poisoned inputs which are labeled correctly and evade such filtering. At this point Turner et al. created

an approach that poisons inputs which appear plausible to humans. The inputs need small changes to make them harder while classifying them but the original label must still remain plausible. This transformation is performed by a GAN-based interpolation and adversarial bounded pertubations. GAN-based interpolation takes each input into the GAN latent space [31] and then interpolate poisoned samples to an incorrect class. Adversarial bounded pertubations use an optimization method to maximize the loss of the pre-trained ML model on poisoned inputs while staying around the original input. The main focus of this attack are poisoned samples that still have poisoned labels. That is why the attack is called Clean Label attack which is originally from [86] in context of targeted poisoning attacks. Figure 17 shows an example airplane with different poisoned samples. The experiments base on the same patterns with a small black-and-white square in the bottom-right corner of the poisoned images as Gu et al. use in their work. The classifier is trained with the poisoned data and the test data are not labeled. The training dataset for the experiments is the CIFAR-10 dataset [48]. The ML model is a trained Wasserstein GAN [6], [35]. A GAN is strategy for training data where a game is defined between two competing networks. It contains a generator network and maps a noise source into the input space. A second network is the discriminator network which receives a generated sample or true data sample and then it must distinguish between those two samples. The Wasserstein GAN is using the Earth-Mover distance which is also called Wasserstein-1. For further explanation of the mathematical structure, please see [36].

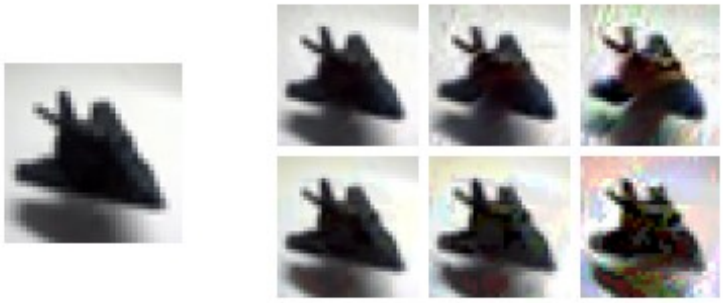

Figure 17: Difference between an original image and the conversion into adversarial examples with different pertubations adapted from [94].

Hidden Trigger Backdoor Attack The last backdoor attack from Saha et al. [79] is the *Hidden Trigger Backdoor Attack* which goal is to let poisoned data look natural with correct labels. This backdoor attack uses a threat model defined from Gu et al. [34]. In this threat model, the attacker provides poisoned training data to a victim that uses it with a pre-trained ML model. The attacker uses a small image as a backdoor which changes the target label to a specific wrong label. This can be used for both attack specificity, targeted and untargeted. With this threat model it is possible to identify the poisoned data because as already explained in *Clean Label Backdoor Attack*, the missclassified images change their label. Identifying the poisoned data in a training set can show how many images are poisoned to measure the extent of possible damage. Saha

et al. propose a threat model inspired from Shafahi et al. [86] and Sabour et al. [78] where the poisoned data are labeled correctly and the backdoor also remains not visible. This is done through optimization for poisoned images which pixel space is close to images from a target category while its feature space is close to source images that are patched with the backdoor. The next part is generalizing the attack for unseen source images which means that the trigger cannot be found during poisoning. Also the trigger should be placed at any random location. In order to implement this, during the optimization the poisoned images pushed close to a cluster of source images that are patched. Based on the work of Moosavi-Dezfooli et al. [61] about universal adversarial examples, Saha et al. minimized the value of loss at all source images and trigger locations. This is done by choosing a random trigger location and source images for each iteration of optimization. Over every iteration a method optimizes randomly patched source images and assigns them to poisoned images closest to the feature space.

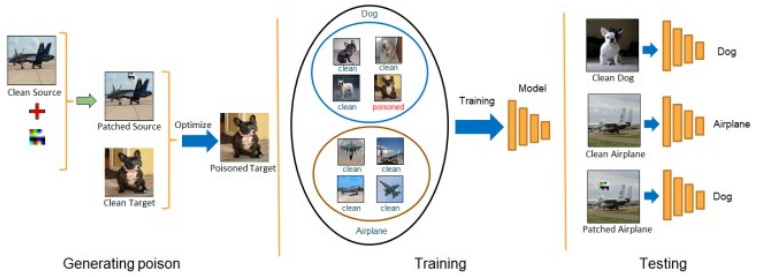

Figure 18: The left side visualize how an attacker generates a set of poisoned images. In the middle the visualization shows how the training data is extended by the poisoned data and then the victim trains the ML model. The right side visualize the test time. An attacker adds the backdoor trigger to images with the source category to manipulate the ML model without changing the label. This visualization is adapted from [79].

3.4. Risk indicators

The selection of the risk indicators is the first part of the RMF to fulfill the requirements. Risk indicators are in context of ISO 27004 [2] attributes and represent the input data. The input data are an object's attributes and assigned to the corresponding measurement method. Breier et al. [18] in subsection 2.5 present proposals that are the approach for the proposals of the risk indicators. These proposals are attack specificity, attack time, attacker's knowledge, and attacker's goal. The proposals have different subcategories which are visualized in figure 20.

Attributes and objects based on ISO 27004

For the RMF the objects are the attack and the attacker. To measure the risks on backdoor attacks, the first step is the selection of objects and associated attributes.

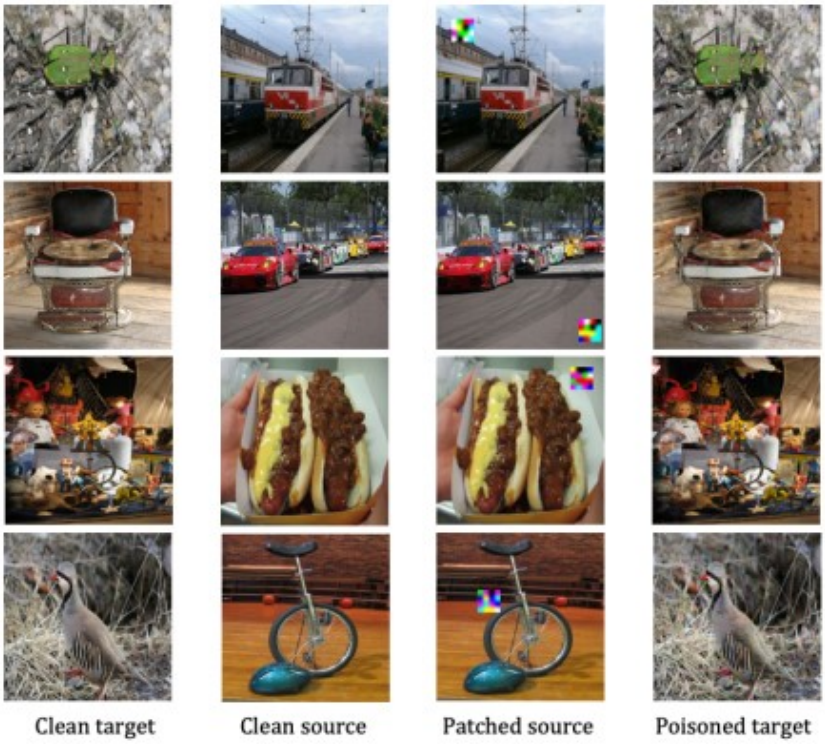

Figure 19: The clean target are images that should be the result of the poisoned images which are the clean sources. These sources are patched with the backdoor trigger adapted from [79].

Beside of the proposed risk indicators defined by Breier et al. this thesis use further risk indicators such as the accuracy of a ML model [30], the computational resources, and the TP, FP, TN, FP [73]. Through their assignments to the two objects they are in turn assigned to the measurement methods which Figure 21 and 22 show. These risk indicators are set out in hypotheses H1 and H2.

3.5. Measurement methods

Risk measurement starts after the identification of suitable objects and attributes 3.4 with the selection of measurement methods according to the requirements and procedure of ISO 27004. The measurement methods start by using backdoor attacks. As already mentioned in the related work section 2, there are different possibilities to execute poisoning attacks. At first, it is important to train a ML model without an attack to get the original values. Xiao et al. [100] describe that training data can be polluted or mislabled when they come from external sources. This means that it is important where the training data comes from to determine, for example, how trustworthy these data are. Furthermore, Turner et al. [94] train a classifier with a small set of clean input data where the input data are images from a trusted source that obtained or inspected the data. So, the more trustworthy a set of training data is, the lower the risk will be. For this thesis

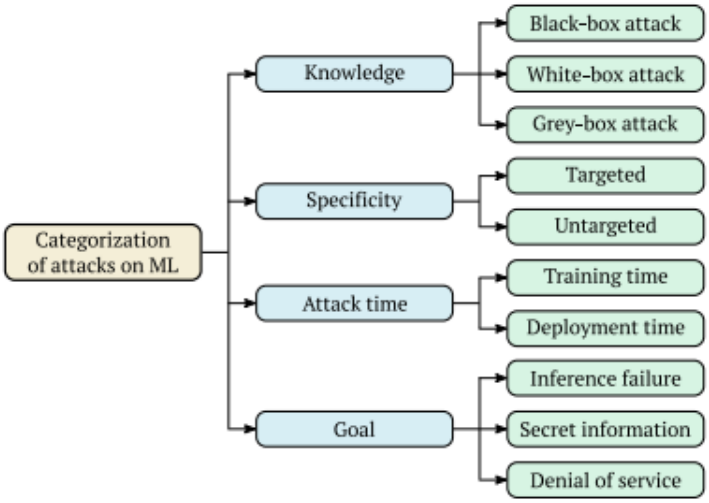

Figure 20: Attack classifications on ML adapted from [18]. The grey-box attack is not observed in this thesis.

this process is skipped and the RMF can not check the origin of the training dataset. The measurement methods expect only what should be measure by the attributes and which attack should be executed on the ML model. A ML model can be outsourced (Machine Learning as a Service, or MLaaS [34]), self-developed, come from an external resource and only has to be trained. For example, Google's Cloud Machine Learning Engine [1] allow's users to upload training data and a TensorFlow ML model to train it in the cloud [34]. Also at this point, no decision is made between these possibilities. Instead of the distinctions, everything is executed locally. This in turn allows to assign all risk indicators at each part in the ML model.
The first measurement method should measure the values to evaluate the extent of damage. The other measurement method is used to find the attacker's effort. Both work according to the threat model of Doynikova et al. [24] by analyzing the data before and after an attack. Therefore the measurement method to measure the extent of damage must be executed before measuring the attacker's effort.

3.6. Measurement method for the extent of damage

Based on the attacks, this subsection explains how the extent of damage is measured. As the hypothesis H1 assumes, the attack specificity, attack time, and the TP, TN, FP, FN are risk indicators. Based on [34], [94], and [79], the accuracy metric of a ML model for the training data is a value that represents the success of a backdoor attack. However, the accuracy is only a total value but does not show in detail the actual extent of the attack. It should be also necessary to compare the original dataset with the poisoned dataset. That comparison would should if the expected number of images are poisoned. Section 4 explains the comparison process with pattern mapping and also a label matching.
All of these risk indicators are assigned with values before and after an attack during the

training of a ML model. These data can be collected directly from the ML model which represents the low-level attributes in the threat model from Doynikova et al. [24]. To get these assigned data, the RMF needs a measurement method based on the low-level attributes. The attack time and attack specificity are both mapped to the high-level attributes as Figure 24 shows. The attack time measures the time an attack needs to be executed in relation to the original training time. The time also differentiates between the training and inference time. The first time measurement is for the attacker's effort while the second part of this risk indicator checks the ML model during training or inference for possible vulnerabilities [77] in the training or test dataset. For example, if the attacker wants to implement a backdoor attack then they have to concentrate on the training dataset for vulnerabilities. That means, the RMF have to check in the training dataset if the backdoor is implemented successfully. If there is a successful match, then the RMF must go through the complete poisoned training dataset to find each difference between the original training dataset and the poisoned dataset. This matching depends on the specificity (searching for the specific target label change or random changes). This process happens After the training, the RMF assigns the values of the risk indicators to the base measures which is the output of this measurement method.

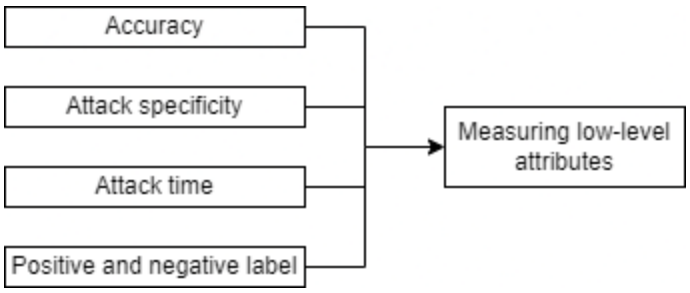

Figure 21: Risk indicators of the attack object in relation to the low-level attributes measurement method.

3.7. Measurement method for the attacker's effort

After explaining the procedure to get the base measures for the extent of damage, this subsection explains how the base measures for the attacker's effort are measured. Breier et al. [18] propose the attacker's knowledge and goal as categories to classify attacks. Further, as described in hypothesis H2, these categories and the computational resources are risk indicators. Computational resources assesses the current use of the central processing unit (CPU), graphics processing unit (GPU), and memory resources to find out what an attacker needs to test and execute their attack. The attacker's knowledge is divided into two possible knowledge levels: black-box and white-box attacks.
The first knowledge level is about black-box attacks where the attacker has no information about the ML model. The second knowledge level are white-box attacks where an attacker has nearly perfect knowledge about the ML model. *Hidden Trigger Backdoor Attack* [79] is a white-box attack where the attacker uses a backdoor trigger on a pre-trained

ML model with finetuning for classification. The backdoor trigger must be implemented while finetuning the ML model to change the label. When the ML model finetunes its training data, the attacker changes the source label to a target label. *Clean Label Backdoor Attack* [94] is a white-box attack which misclassifies during training time and not during inference. With this goal, the images should be harder to be correctly classified and the ML model should recognize the backdoor trigger as a feature. This procedure should prevent modifications during training time but it is used exactly for that purpose. This attack is implemented based on projected gradient descent (PGD) [57]. PGD is an approach to train NN with an improved resistance against adversarial attacks. *Pattern Backdoor Attack* [34] is a black-box attack which adds a pixel pattern or single pixel into an image and misclassifies these poisoned images to a target or untargeted label. After adding the pattern, these poisoned images have to be mixed back into the original training dataset. All of these black- and white-box attacks are useful for the RMF to represent an attack scenario for the risk measurement.
After one of these attacks are executed, it should be possible for the RMF to measure the knowledge by pre-determined steps to execute the attack [29]. Pre-determined steps originate from common criteria which this thesis get from the explained attacks. Every attack has its own steps during the development and execution. The attacker's knowledge is the risk indicator that count these steps.
The attacker's goal is the last risk indicator to measure the attacker's effort to find out what the attacker is trying to achieve [18]. To find this goal in the measurement method, this risk indicator has to get the attack type (i.e. backdoor attack) to return the goal and count the steps it takes for the development. If one of the previously mentioned attacks is identified, the goal is to measure pre-determined steps too. The pre-determined steps in the attacker's knowledge and goal are also summarized with the low-level attributes which Figure 24 shows. When all risk indicators are assigned with its values then the measurement methods can pass all results to the base measures.

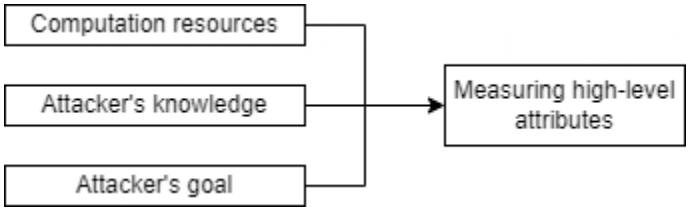

Figure 22: Risk indicators of the attacker object in relation to the high-level attributes measurement method.

The high-level attribute measurement method should return all values to measure the attacker's effort. These values are only able to represent the attacker's effort if they are combined with specific values from the low-level attribute measurement method. This mapping method explains the following subsection.

3.8. Using the formal threat model

To measure the extent of damage, subsection 3.6 describes which risk indicators the RMF uses to get the corresponding values. As already explained in subsection 3.6, these values must be collected from the measurement method of the low-level attributes which will be explained in this subsection. After that, subsection 3.7 explains how to get the corresponding values for the attacker's effort. The concept to measure the corresponding values that are assigned to the risk indicators are described in subsection 3.7 and is further explained here by the threat model. In this subsection, the research questions RQ2 and RQ3 are addressed in more detail. In reference to the research question RQ4 this threat model is a possible method for the RMF to measure risks which is explained in section 5. As discussed in section 2.3, the high- and low-level attributes have to be mapped which is also explained in this subsection.
Before explaining the procedure, there are two requirements for the dataset that is collected from the attributes [24] for the analysis process. These requirements are already transferred to ML models:

1. The first requirement is a dataset which contains information about the attack actions against a ML model. The information must be based on the skills, resources, intention, and motivation of the attacker.

2. The second requirement for the dataset is that everything is marked in such a way that the analysis shows which actions the attacker performed. However, this requirement is more about having multiple attackers or the analysis across multiple attackers.

The second requirement can not be fulfilled because this thesis does not differentiate between one or multiple attackers. There is always just one attacker considered in this thesis.

The low-level attributes

To find the extent of damage, there is a need to collect data which can be measured from every attack on a ML model. These data are classified and explained in subsection 2.3. Doynikova et al. [24] explain which data are required to measure the low-level attributes. The low-level attributes that Doynikova et al. use in their work are event logs and network traffic that are possible to use in relation with the risk indicators. A possibilty to get the risk indicator values is monitoring them with event logs. The network traffic can not be used because the RMF only measures an attack on a ML model locally that does not include MLaaS [37].
The low-level attributes measure the extent of damage based on the collected data. It is therefore important that data about the ML model and the attack can be collected during the entire training and inference process. For example, if the attack specificity needs to be measured then it must be possible to get this risk indicator through the attack itself.

After the assignment of the values, Doynikova et al. explain that the low-level attributes must be mapped to the high-level attributes to measure the attacker's effort. The high-level attributes are delimited, unlike the low-level attributes, which means the attributes must be evaluated in a separate measurement method afterwards.

The high-level attributes

The high-level attributes measure the values for the attacker's effort based on the risk indicators such as attacker's knowledge, attacker's goal, and the computational resources. The following four groups explained in subsection 2.3 are placed in relation to the risk indicators. The first group includes characteristics such as skills, motivation and intention. The motivation and intention correspond to the attacker's goal. The skills are a characteristic that are represented in the attacker's knowledge and goal. The second group characterizes the attacker's capabilities [102] and show the characteristics as the computational resources. This risk indicator measures which hardware an attacker needs to execute an attack. The third group incorporates the attacker in relation with the attacked system. This group includes the attacker's location, the privileges, his goals, the access and the attacker's knowledge. This part can be used for the RMF with the attacker's knowledge and goal. The attacker's location and privileges are not part of the risk measurement. The last group relates the attacker with the attack and the steps that are included to execute the attack. This group can be used to map the low- and high-level attributes which is explained below.

3.9. Define measurement functions to calculate derived measures

In this subsection the base measures are also called risk indicators because the RMF shows the output of the measurement methods assigned to the respective risk indicator. Not every derived measure needs to be calculated. The TP, TN, FP, and FN can be calculated in a measurement function to different ML metrics such as the accuracy [63], confusion matrix [101], precision-recall [23], and F1-Score [63] which subsection 2.4 describes. This risk indicator can be also used to measure the extent of damage because each true and false value represent the development of the ML model during the training time. These metrics are used to show the differences between an original and poisoned ML model. The derived measures that represent values of the ML model are only calculated from one risk indicator because this risk indicator has multiple different values that are needed to calculate the ML metrics.
The risk indicators which are combined to derived measures are shown in Figure 23. The first derived measure in Figure 23 describe where and how to execute it. If all of them are combined, then should the derived measure be a value that represents without the computational resources how high the effort is to implement, test, and execute an attack. The measurement function bases on the mapping from Doynikova et al. [24]. Doynikova et al.'s mapping transfers measured values from the attack to evaluate the attacker's effort which is explained in subsection 31.

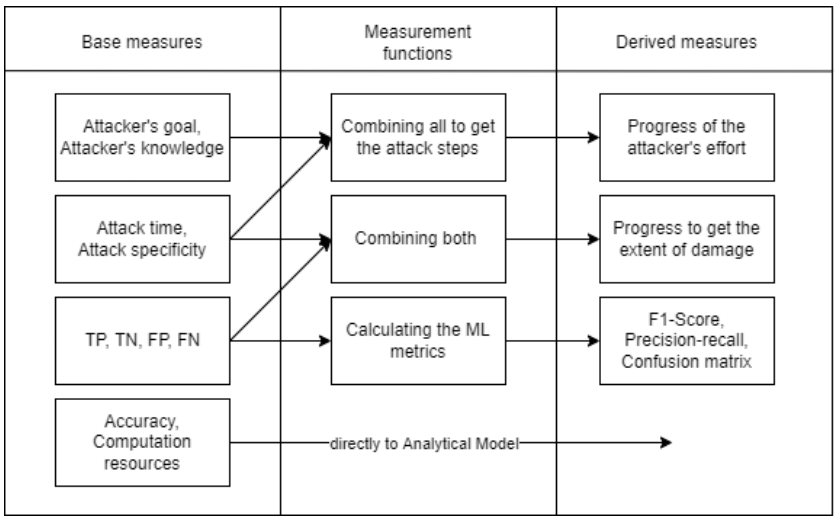

Figure 23: Combine the risk indicators to derived measure and evaluate the derived and base measures with the analytical models to indicators.

3.10. Mapping the low-level with the high-level attributes

After explaining the low- and high-level attributes as separated measurement methods and measurement function requirements, they must be mapped based on the threat model of Doynikova et al. [24]. At this point, the attacks are displayed on the basis of their values. At the beginning, the data of the attacks are measured separately in the measurement method of the low-level attributes and then those of the high-level attributes. The data here always refer to the risk indicators before and after an attack to monitor the original and the manipulated training dataset.
The combination starts with the attack time and specificity where the attack is executed during training or inference time. If the attack is, for example, executed during inference time, then the concentration is located on all steps after the training of the ML model. If the attack specificity is untargeted, then the attacker can poison the training dataset without finding out label names and only has to add a backdoor trigger. When the attacker decides to implement a backdoor attack, then they have to go through pre-determined steps which are defined depending on the used attack. The same applies to the goal which is based on pre-determined steps too. With these information it is possible to combine the steps and then getting the derived measure. The same procedure combines the measurement function for the extent of damage from the attack time and specificity. The accuracy and computational resources are not considered further here, because they are transferred directly to the analytical model.
The accuracy is not considered for the mapping because the value is only relevant to measure the extent of damage. The attack time and specificity is relevant for the extent of damage and to find the attacker's effort. These risk indicators get values that can be transferred to the risk indicators such as attacker's knowledge and goal. The computational resources are a risk indicator that is only for finding the attacker's effort. Figure 24 summarizes this mapping and in comparison to the threat model of Doynikova

et al., the values are collected and evaluated completely without human judgment.

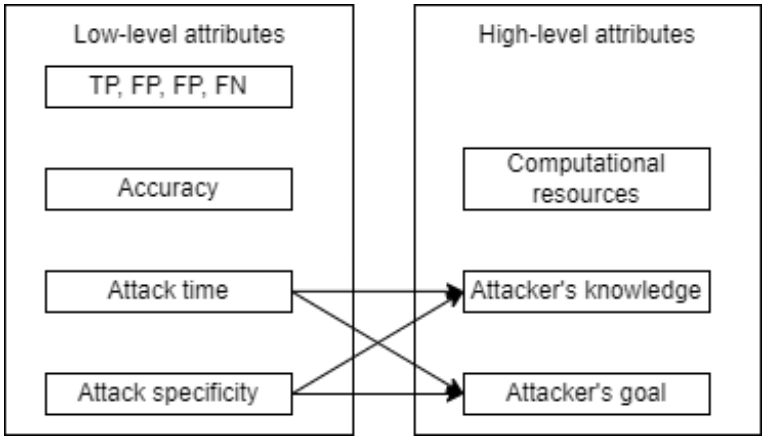

Figure 24: Mapping between the low- and high-level attributes.

3.11. Define analytical models and indicators

The indicators are the final results from the measurement and provide the transition over the decision criteria to the measurement results. After evaluating the derived measures, all measures have to be combined for calculating all values to get the extent of damage and attacker's effort. To get both values, the RMF needs to combine the base and derived measures depending on the target value which Figure 25 shows. Resulting from the combination, it is necessary for this thesis that the RMF creates two indicators. Depending on that requirement, there have to be two analytical models [2]. At this point, the analytical models have to depict functions to calculate the indicators for the interpretation in the decision criteria.
The first analytical model combines all measures for the extent of damage. The second analytical model combines all measures to get the attacker's effort. Both analytical models caculate the values by addition. This addition is intended to represent each individual risk indicator as a total value.

The transition between measurement and measurement results

After the indicators are calculated, then the next step is the transition to the measurement results by interpreting the indicators based on the decision criteria. At this position it is important to relate the attacker's effort with the probability of occurrence, because this probability is used to get the final risk value.
The goal for the decision criteria is to interpret the indicators without human judgment. Based on this interpretation, the functionality of the decision criteria are crucial for how the indicators can be interpreted for the given intervals. The interpretation should make it possible to find the main factor by human judgment that increases the risk value after the risk measurement.

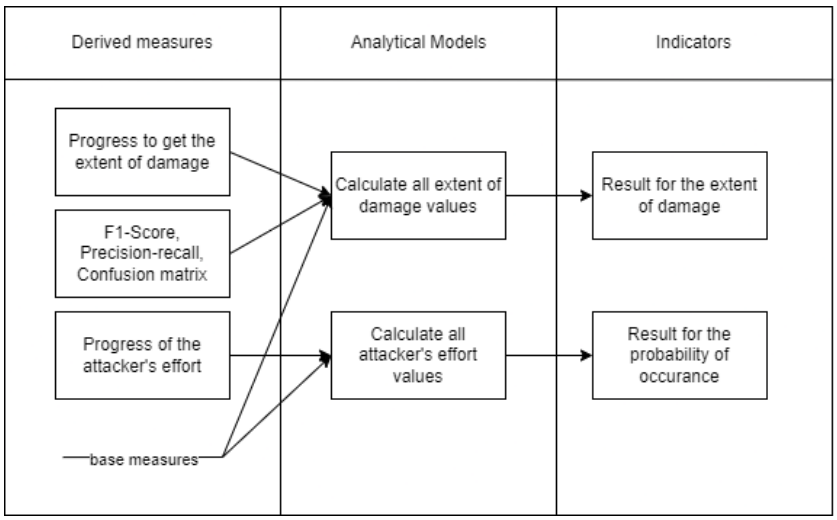

Figure 25: Calculation of the derived and base measures in the analytical models to get the indicators.

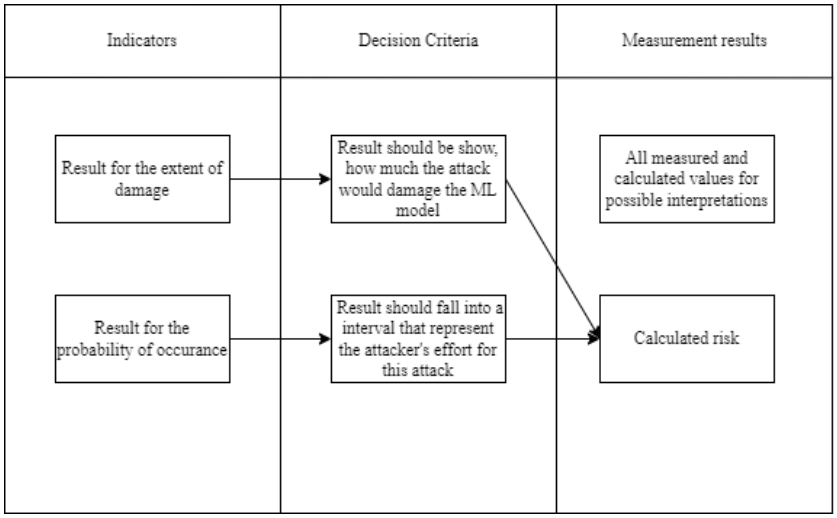

Figure 26: Interpreted results that should contain requirements which specify the decision criteria.

3.12. Develop measurement results by evaluating the risk measurement

The implementation of the measurement results are possible through the decision criteria which are also defined in [2]. The RMF shows the results as visualized Python plots and calculated results. Both are further explained in section 4.

Machine learning metrics for risk measurement

With regard to poisoning attacks, the goal is to decrease the accuracy [16], [34]. But the RMF should also use the confusion matrix, precision-recall, and the F1-score of the training process to show the differences of the training with the original and poisoned training dataset. With regard to the attacker's effort every collected information of the ML model and the training data could be a possible value.

Calculate the risks

The main calculation is *Risk* = *Extent of damage* ∗ *Probability of occurrence* [43]. This calculation is intended to show how high the risk of an attack is for a ML model.
Now it must be clarified how the probability of occurrence and the extent of damage are represented as values. The probability of occurrence bases on the attacker's effort. This thesis assumes that the higher the effort, the lower the probability of occurrence. The value is represented by a non-negative natural number $x \in \mathbb{N}$. This depends on the number of steps an attacker must perform to execute the attack. The extent of damage is also calculated by the addition of all measures. E is the extent of damage, A is the attacker's effort, n the number of risk indicators, and r the risk indicator.

$$\mathrm{E} = \sum_{i=0}^{n} \mathrm{r}_i$$
$$\mathrm{A} = \sum_{i=0}^{n} \mathrm{r}_i$$

The addition of all risk indicators to one value should show the total value of each indicator. This relation between the *Risk* and risk indicators should then be depicted in a risk matrix. For example, if the accuracy is closer to the accuracy of the original ML model then the *Risk* could be still in a red field as Figure 27 shows but the value would be in another field as if the computational resources were closer to the values of the original ML model.
This *Risk* is classified in a matrix to represent the attack on the ML model. Figure 27 shows a sample matrix where green is the lowest risk level and red the highest. The *x-axis* and *y-axis* should show different combinations to classify the values that classify the risk value into the risk matrix. This combinations cannot be shown because there are no standards or scientific works which specify it.

This classification bases on the European Telecommunication Standards Institute (ETSI). In their standard [82], ETSI show a risk classification standard which table 1 shows.

3.13. The implementation plan of the RMF

The last point (g) - "Establishing measurement implementation approach and documentation" of 3.1 shows the needed information for an implementation plan. This subsection shows the final concept as a complete structure.

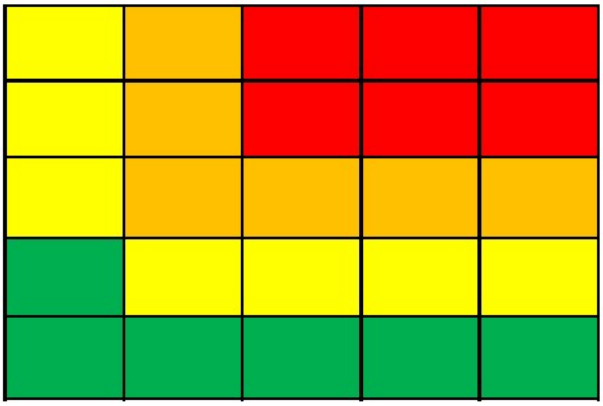

Figure 27: Sample risk matrix adapted from [41].

Risk	Explanation
Minor	"No essential assets are concerned, or the attack is unlikely. Threats causing minor risks have no primary need for counter measures."
Major	"Threats on relevant assets are likely to occur although their impact is unlikely to be fatal. Major risks should be handled seriously and should be minimized by the appropriate use of countermeasures."
Critical	"The primary interests of the providers and/or subscribers are threatened and the effort required from a potential attacker's to implement the threat(s) is not high. Critical risks should be minimized with highest priority."

Table 1: Risk classification adapted from [82]

The final concept and design for the RMF

After explaining the expected results, the last part of this section shows the summarized concept and design to implement the RMF in the following section. Table 2 and 3 show the two objects with its attributes that represent the risk indicators for the measurement methods afterwards.

In Figure 28 the identified risk indicators are passed to the measurement methods which are one of the two main components for the risk measurement. To measure the extent of damage, the low-level attributes assign the values of the attributes from the attack object. To get the attacker's effort another measurement method should represent the measurement of the high-level attributes. These two classes need to be mapped. Next, the backdoor attacks must be selected to measure the value that is assigned for the next main component, the measurement which follows the requirements from ISO 27004 [2].

Base measures are all output values from the measuremen methods. These measures,

Attack object	
Attributes	Description
Accuracy	The accuracy relates the number of data examples with true predicted labels to the number of all examined data examples [63]
TP, FP, TN, FN	This risk indicator can be used to calculate different ML metrics and show its values during the training time
Attack specificity	The attack specificity can be targeted or untargeted
Attack time	The attack time differentiate between training and deployment time

Table 2: ISO 27004 Object (attack)

Attacker object	
Attributes	Description
Attacker's goal	The attacker's goal can be a denial-of-service, doing an inference failure, or obtaining secret information
Attacker's knowledge	The knowledge of an attacker can be categoraized between white-, black-, or grey-box
Computational resources	The CPU, memory, GPU are the three parts which represent the used effort from the computer to attack and train a ML model.

Table 3: ISO 27004 Object (attacker)

then has to be splitted into base measures for as the input for the measurement functions and the measures that the analytical model takes directly. Each measurement function calculate a derived measure to get the attacker's effort, the extent of damage in the further measurement process and a function to get the ML metrics to show the differences before and after the attack. The derived measures and base measures which are used for the analytical models directly, calculate the final attacker's effort and extent of damage. The indicators which are the output of the analytical models are determined on the basis of the decision criteria if they can be used to calculate the *Risk* [43]. If the indicators pass the decision criteria then they can calculate the *Risk* and be classified in the risk matrix [41]. For further human judgment the measurement results show also the ML metrics and all assigned values to the risk indicators.

3.14. Expected results of the RMF

After attacking a ML model without considering the defense, the values of the accuracy of the poisoned data should be the lowest value (%) as possible, which is the highest effectiveness of the attack. The accuracy of the original training and testing should be completely unchanged or so minimally changed that it is not noticeable. The attack

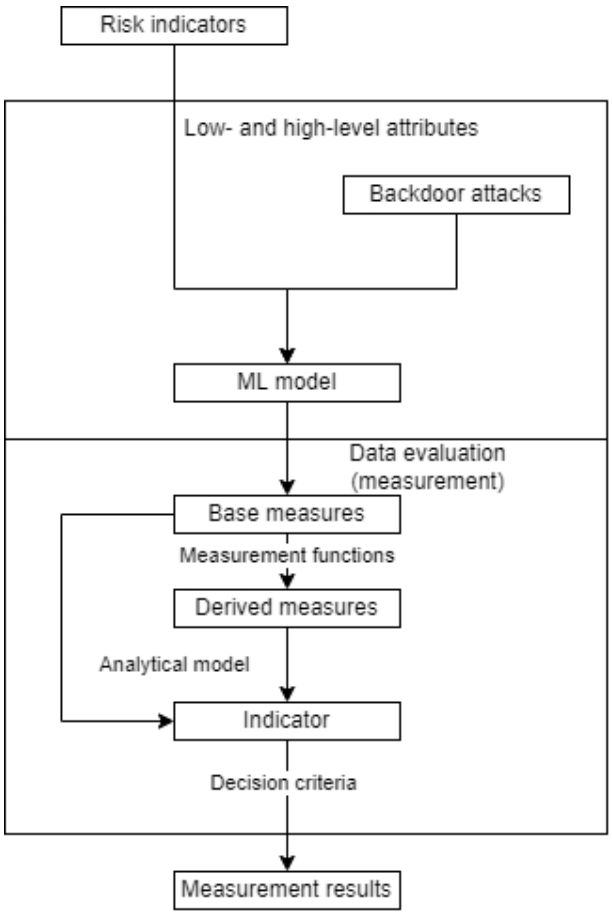

Figure 28: Complete concept architecture adapted from [2].

time should always return, that the attack is executed during training time because all attacks in this thesis are backdoor attacks [95]. Based on this result, the attack time should always find backdoors in the poisoned training dataset and the execution time should be longer than the original training time. The attack specificity should be always measure if an attack is targeted or untargeted, if the attacker specifies a target or not. Attacker's goal should always return an inference failure goal [18]. Attacker's knowledge should measure the attacks depending if it is a white- or black-box attack, which attack time, and attack specificity is measured before. The risk result of the risk measurement should be not possible to classify in the risk matrix because there is no scientific work which classify risks in ML security. In conclusion, it is not possible to expect that the evaluation of the risk measurement can say how high or low the risk is.

4. Implementation

Afer explaining and discussing the concept this section describe the implementation of the technical framework RMF by following the procedures of ISO 27004 [2]. The technical RMF uses Python 3.7 [97] as the programming language and ART as the basis. Beside the attacks given by the ART, there is a function from the technical RMF to execute individual attacks. This technical RMF should be used a step ahead of using the framework of Schwerdtner et al. [84].

4.1. Structure of the RMF

Directory tree

The RMF is structured as follows:

```
rmf/
├── attacks/
│   └── art/
│       └── backdoors.py
├── backdoors/
│   └── png-Files
├── measurement/
│   ├── monitoring.py
│   ├── log.py
│   └── measurement.py
├── visualizations/
│   ├── data_output.py
│   └── plot.py
├── log_file.log
└── case_study.py
```

The *png-Files* are used by the *backdoors.py* script to add patterns into the images that are the backdoor triggers. Both folders *measurement* and *metrics* contain implementations to monitor the collected data for the risk measurement and evaluate it. The *visualizations* folder shows the measurement results and summarize them into one output file.

4.2. Implementing the risk indicatiors

The risk indicators are the main part for the risk measurement. Therefore the risk indicators are used through the complete risk measurement. Every risk indicator is implemented as an own function in the RMF. This makes it possible to measure the risk indicator values at every step during the ML model training. The goal is to use measure the risk indicator values with the original training data and the manipulated training data. The $accuracy = \frac{n(correctly\_predicted)}{n(all)}$ is implemented as Nguyen and Zeigermann [63]

describe it. The next risk indicator is the attack time which checks the time an attack needs to be executed and if the attack is executed during training or inference time. Depending on the targeted phase, the function for this risk indicator checks the ML for vulnerabilites. Every Vulnerability increases a counter to check how much exploits are in the ML algorithm. Since in this thesis only backdoor attacks are performed, finding vulnerabilities refers to checking if backdoor triggers are in images [34], [55]. This matching of patterns in images do the Fast Library for Approximate Nearest Neighbors (FLANN) [28] in relation with the Scale Invariant Feature Transform (SIFT) algorithm [90]. The following Python algorithm shows how both work in combination:

```
img1 = cv.imread('box.png',cv.IMREAD_GRAYSCALE)          # queryImage
img2 = cv.imread('box_in_scene.png',cv.IMREAD_GRAYSCALE) # trainImage

# Initiate SIFT detector
sift = cv.SIFT_create()

# find the keypoints and descriptors with SIFT
kp1, des1 = sift.detectAndCompute(img1,None)
kp2, des2 = sift.detectAndCompute(img2,None)

# FLANN parameters
FLANN_INDEX_KDTREE = 1
index_params = dict(algorithm = FLANN_INDEX_KDTREE, trees = 5)
search_params = dict(checks=50)   # or pass empty dictionary
flann = cv.FlannBasedMatcher(index_params,search_params)
matches = flann.knnMatch(des1,des2,k=2)
```

img1 is the pattern that the algorithm should find in *img2*. SIFT is feature extraction algorithm which detects points of an image's scale space. That detection comes from a variable-scale Gaussian convolution that takes an input image. The next part is the positioning of feature points, then the algorithm assigns the direction of these feature points with a pixel gradient histogram. The last part of the algorithm is a description of each feature point's uniqueness by a feature vector. After SIFT the FLANN algorithm searches based on Approximate Nearest Neighbor (ANN) search [62] with a KD-Tree [15]. After each match the RMF uses the prediction function of Keras to check if the poisoned image predicts the correct target label and then saves the correct label and expected target label as a key-value pair to count how many images are missclassified correctly. This label relation should evaluate in more detail the extent of the attack on the ML model.
Attack specificity is an risk indicator that measures if the attack is targeted or untargeted. The ART attacks have a parameter that takes a target optionally. If the target is random, then there is no specific label. If it is targeted, there is a specific label where a certain number of images in percentage are poisoned or specific images of a targeted label. As next is the TP, TN, FP, FN. Keras show these values with a implemented function in this technical framework. The computational resources get the values from the operating system for the specific executed Python script. The attacker's knowledge is implemented based on the attack. For example, if the *Pattern Backdoor Attack* is implemented in the

ML model, then the function for the attacker's knowledge returns the steps to implement this specific attack [29]. Attacker's goal [18] can be an inference failure, obtain secret information, or executing a denial-of- service. The goal of this risk indicator is to evaluate it based on monitored data of the attack. The attack must have characteristics that represent the goal an attacker has. An inference failure happens during the testing phase. So when an attacker try to execute an inference failure he could use exploratory attacks [92] or evasion attacks [19]. Obtain secret information can be for example stealing model attacks [99], [104]. Denial-of-service attacks have the goal to slow down the ML model or crash the ML model completly [96]. If one of these goals is identified then this function returns pre-determined steps on how to achieve this goal. If an attacker tries to missclassify the ML model based on backdoor attacks, then the goal could be an inference failure because the image classification fails during inference time. To achieve this goal the attacker has to develop a backdoor attack. The difference between achieving this goal and getting the knowledge of the attacker is about how they develop the backdoor attack to get what he wants and how much he knows to implement the attack into the ML model. The goal do not concentrate on specific attacks, only on what the attacker have to do in general.
Subsection 4.3 explains how the attacks in the measurement methods are implemented and what an attacker have to do to implement them into a ML model. After that, subsection 4.4 explains how to get the risk indicator values if an attack is executed.

4.3. Implementing backdoor attacks from the ART

The following three attacks are all based on the ART [66] and represent white-box and black-box attacks and both targeted and untargeted attacks. These attacks are the main factor to assign the values to the base measures. The attacks already differentiate between the original and poisoned dataset for evaluation. To implement the attacks and measure the attacker's goal to get an inference failure, the following steps has an attacker to do [80]:

1. Taking one or more backdoor pattern
2. Taking target labels corresponding to the patterns
3. Place it at a specific or random location
4. Place the backdoor trigger to a subset of the original data
5. Mix it with the original data
6. Mapping the backdoor triggers to the corresponding target labels

Backdoor attacks from the ART

All backdoor attacks use the *Pattern Backdoor Attack* [34] class which expects a pattern argument. The pattern is a picture which is for the poisoned images in the training

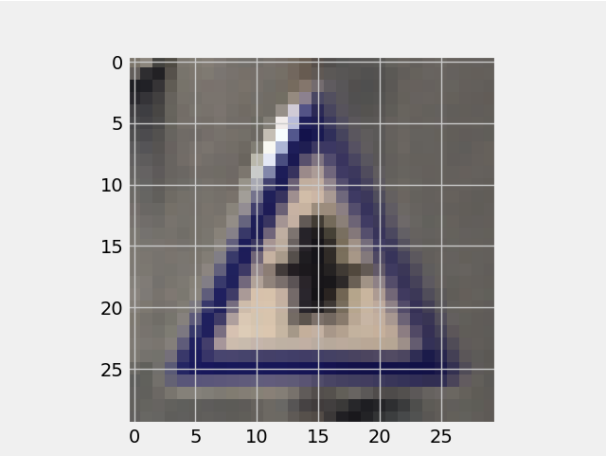

Figure 29: Street sign without a backdoor pattern with the label *Right-of-way at intersection.*

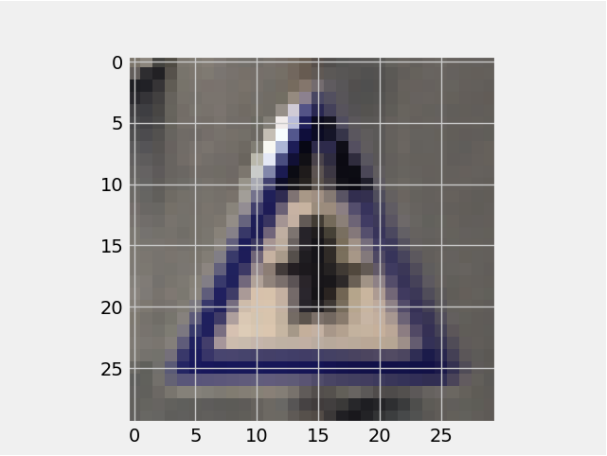

Figure 30: Street sign with a backdoor pattern but still with the original label *Right-of-way at intersection.*

data. The pattern must be implemented before training and choose a random selection of images. This is implemented in the RMF as follows:

```
n_train = np.shape(x)[0]
num_selection = num_of_rand_images
random_selection = np.random.choice(n_train, num_selection)
x = x[random_selection]
y = y[random_selection]
```

Then the arguments x and y are passed to the poisoning function from the ART. x and y are parameters which the function expects when calling it. Appendix A describe the functions and parameters. Further it is important that the shape of the images from the training data are *N, H, W, C* or *N, C, H, W*. N is the number of images in the batch, H is the image height, W is the image width, and C is the channel number of an image such as grayscale or RGB. After adding a backdoor as a pattern to the random images, the poisoned images are replaced back to the training data. These poisoned images are saved in a different folder to check if the images are missclassified after or while testing the ML model.

The first attack uses the *Pattern Backdoor Attack* class without any other backdoor classes from the ART. This attack which bases on Gu et al. [34] is a black-box attack which is untargeted. In the RMF it uses a pattern backdoor which Figure 29 and Figure 30 show as the difference between the original and the poisoned image. The goal of this attack is to change the original label to a random other label. This attack can be executed without any information because it is not important which training data the ML model use and what ML model is used for the training itself.

After poisoning the images these image need be copied back into the original training data. Before this can be done the poisoned data must replace the original training data

which contain no backdoor. When the RMF take the random original images, it save them into a temporary variable and then delete the images from the original training data. The next step is the poisoning while the poisoned data and the original training data need the same shape and dimension which make it possible to copy the poisoned images. As mentioned before the poisoning function takes four specific shapes which make it easy to have the same shape in the original training data and poisoned data. The implementation of the attack additionally shows the effort an attacker have to implement this attack. Thus the steps can be used for the attacker's knowledge risk indicator. The steps to execute this attack are the following:

1. Choose a backdoor pattern
2. Take number of random images to poison
3. Add the backdoor pattern to the images
4. Mix poisoned images back to the original dataset

The *Clean Label Backdoor Attack* [94] poison images and misclassify during the training time. To use this attack the *Pattern Backdoor Attack* must be used before and then the clean label attack can be executed after training. After adding the backdoor trigger with *Pattern Backdoor Attack* the original training data are trained with a proxy classifier which should do the same classification tasks as the original classifier. The first poison step is selecting target images. As next the PGD (which is untargeted) must be implemented to make it harder classify correctly. The last step is adding the backdoor trigger and add the target label. The steps to execute this attack are the following:

1. Choose a backdoor pattern
2. Select a source image
3. Select a target image
4. Reverse engineer the classifier
5. Select a learning rate
6. Select the number of maximum iterations
7. Select the number (%) of images to poison
8. Poison the original training dataset
9. Train the original dataset with the proxy classifier
10. Train the poisoned dataset with the original classifier

The *Hidden Trigger Backdoor Attack* [79] need to be executed after training. To add the backdoor, the *Pattern Backdoor Attack* must be used before. After training, the poisoned data and a smaller number of clean training inputs are used to finetune the model. Finetuning means, adjusting parameters of the ML model to improve it with a small amount of training data to improve the performance [52], [22]. For the attack the poisoned data are used with a small amount of clean training data. The following Python code shows the finetuning of a ML model where the attack is implemented.

```
dataset_size = size
num_labels = label_size
num_per_label = dataset_size/num_labels

poison_dataset_inds = []

for i in range(num_labels):
    label_inds = np.where(np.argmax(y_train, axis=1) == i)[0]
    num_select = int(num_per_label)
    if np.argmax(target) == i:
        num_select = int(num_select - min(num_per_label, len(
 poison_data)))
        poison_dataset_inds.append(poison_indices)

    if num_select != 0:
        poison_dataset_inds.append(np.random.choice(label_inds,
 num_select, replace=False))

poison_dataset_inds = np.concatenate(poison_dataset_inds)

poison_x = np.copy(x_train)
poison_x[poison_indices] = poison_data
poison_x = poison_x[poison_dataset_inds]

poison_y = np.copy(y_train)[poison_dataset_inds]
```

The parameters of the *Hidden Trigger Backdoor Attack* are analog to the *Clean Label Backdoor Attack* but the pattern is different as the backdoor trigger which Figure 19 shows.

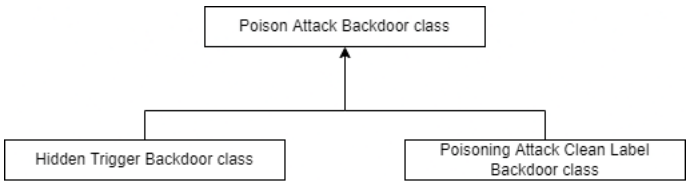

Figure 31: Both attack classes (*Clean Label Backdoor Attack* and *Hidden Trigger Backdoor Attack*) get the backdoor from the *Pattern Backdoor Attack*.

The steps to execute this attack are the following:

1. Choose a backdoor pattern

2. Select a source image
3. Select a target image
4. Add the backdoor to the images
5. Finetune the ML model with the poisoned dataset
6. Select original training images for the finetuning
7. Mix the poisoned with the original images

4.4. Implementing the threat model

The measurement methods are implemented based on the threat model of Doynikova et al. [24]. This thesis concentrates on specific risk indicators whereby the threat model functions do not expect risk indicators as arguments. The low- and high-level attributes do not need an own function because both can be saved into a list of values. So, there is only the mapping function which takes these values as arguments.

Mapping between the attributes

The attack time and specificity are the only risk indicators which have to be mapped to the attacker's knowledge and goal. Based on the low-level attributes, the high-level attributes are additionally adjusted after the measurement. After the execution of the attacks, the values of the attack itself - specificlly the attack time - is used to measure the time an attacker needed to execute its attack [18]. This measurement is based on the difference between training of the ML model without and with the attack. The attack time for measuring the extent of damage is used to see which vulnerabilities a ML model has during the training or test time [77]. Because values are measured that can be used to measure the effort of an attacker, this risk indicator must be mapped.
For the attack specificity the RMF measures how much images are poisoned and what the attacker has to do to poison targeted images [106] or how he poison random images [34]. The amount the attack has to do to poison images is mapped into the attacker's goal and knowledge.
The following subsection describe how the results from the measurement methods are evaluated in the measurement construct.

4.5. The implemented measurement construct

After assigning the risk indicators based on the threat model of Doynikova et al. [24], this subsection explains the implemented measurement construct starting from the base measures up to the decision criteria.
The measurement function is a single function which takes the base measures that has to be combined to the derived measures. To access these base measures, they must first be separated from the base measures that are given directly to the analytical model function.

Figure 32 visualizes the procedure of the measurement function. The first measurement function calculates the derived measures of the TP, TN, FP, FN into the ML metrics such as F1-Score, precision-recall, and confusion matrix with the mathematical functions from Nguyen and Zeigermann [63]. These ML metrics show the total results of the ML model's performance and represent as total results how the attacked changed their values. Since the value of the *Risk* should represent a higher risk at a high value than at a low value, the base measures as well as derived measures must output corresponding results. In order to ensure this representation, it is important for the accuracy, F1-Score, precision-recall, and confusion matrix that their reduced values due to the attack are used as a high value for the risk. Since the accuracy and the other ML metrics are represented as percentages and thus lie between 0 and 100% or 0 and 1, the RMF reverses these values. If the accuracy is for example 0.06 or 6%, the RMF converts this value to 9.94 out of 10. This allows the value to appropriately reflect how much the accuracy increases the overall *Risk*. To calculate the derived measure for the extent of damage the last value calculate how much labels are missclassified. This matching is implemented by key-value pairs whereby the key is the original predicted image and the value is the poisoned predicted image. If the poisoned prediction is the correct missclassified target label then a counter increases. Then the result of this matching can be compared with the possible number of poisoned images to check how much images are actually poisoned. The resulting percentage reflects how effective the attack was on the tested ML model. In order not to depend on the different ML libraries the RMF gets its own functions of the different ML metrics. That increases the support of different Python libraries for ML risk measurement. The next function in Figure 32 combines the attack steps by adding all steps together.

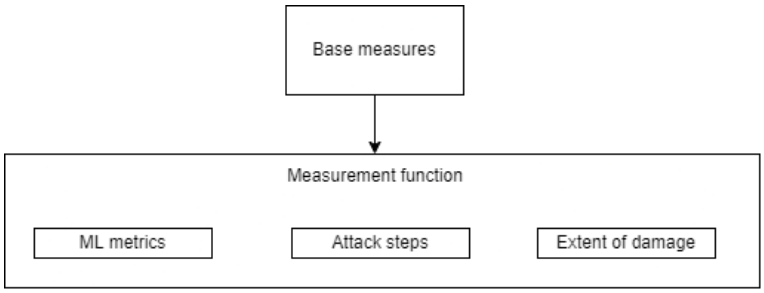

Figure 32: The measurement function takes the base measures and distribute them to the inner measurement functions.

Figure 33 shows the procedure of the analytical model. The analytical model takes the base measures that are not calculated into derived measures and the derived measures from the measurement function. All values for the risk calculation are combined here and calculated to get the probability of occurrence and the extent of damage.

The last step to get the measurement results shows Figure 34 which input for the decision criteria are the two possible indicators from Figure 33. The attacker's effort have to be in an interval which have to be specified as an information need. The extent of damage must also be checked based on an information need [2]. For this thesis there

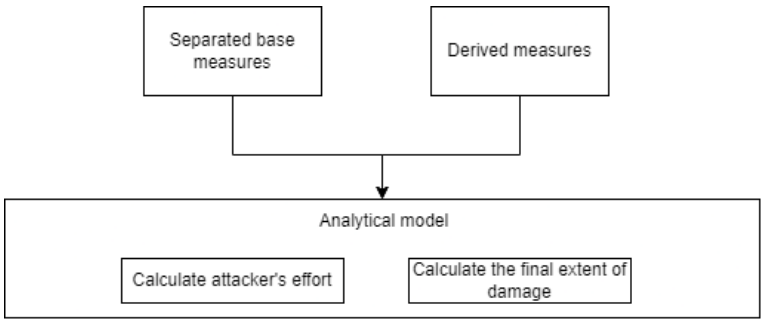

Figure 33: The analytical model takes the separated base measures and derived measures. Then it distribute them to the inner analytical models.

is no interval because the information need is not considered further. If the criteria is fulfilled the indicatos are able to be calculated and then the result can be depicted. This procedure describe subsection 4.7.

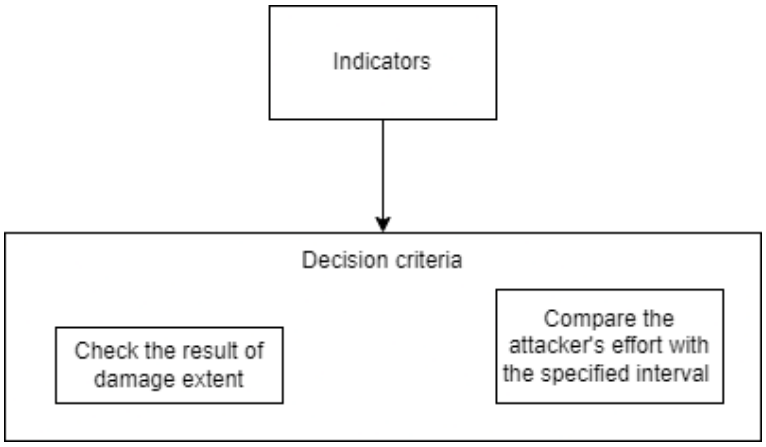

Figure 34: The analytical model takes the separated base measures and derived measures. Then it distribute them to the inner analytical models.

4.6. Evaluation Method

The following steps describe the complete process on how the *Risk* can be depicted in the risk matrix (the flow graph depict in Figure 35 the process of this evaluation) [18]:

1. All weights are chosen based on result's impact of each risk indicator on a ML model.

2. Each value of a risk indicator is assigned to a weight which represent the impact on the extent of damage or probability of occurrence.

3. The value of a risk indicator is mapped with the identified weights.

4. The weights depict the structure of the risk matrix.

5. Then the calculated *Risk* is depict in the risk matrix.

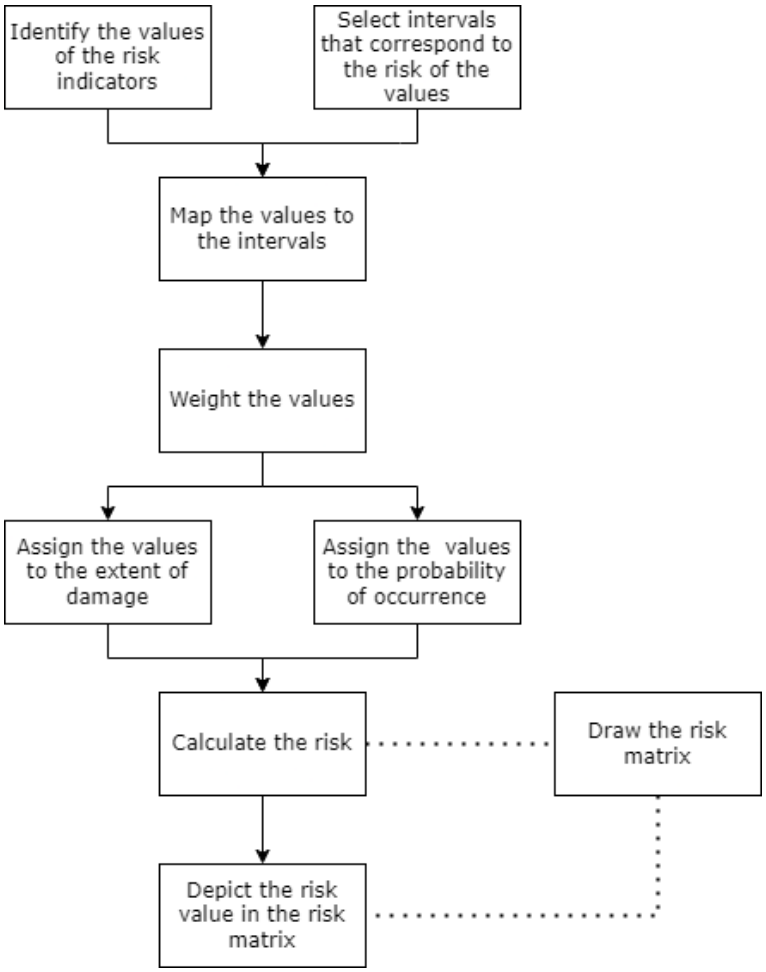

Figure 35: The process to depict the *Risk* on the risk matrix.

The goal of this flow graph is to weight the *Risk* based on the original ML model. For example, if the accuracy is 90% then this value represents the best approximation for the *Risk* value in the risk matrix. The same weighting applies for the other risk indicators which are able to differentiate between the original and attacked ML model. This weighting could make it possible for an organization to decide the lowest and highest *Risk* based on the output data and to create the decision criteria.

4.7. Show the measurement results

The measurement results show the monitored data based on the measurement process of ISO 27004 [2]. This contains all logged risk indicator values and visualized ML metrics. A second part shows the risk for the ML model with an attack. The risk is calculated

by *Risk = Extent of damage ∗ Probability of occurence* [43]. This risk value is depicted in a risk matrix which shows how high the risk is with the executed attack. Based on the measurement result should it be possible to interpret the attack but the RMF shows no defense methods. The RMF concentrates only on the attack and the associated risk measurement. What happens with these results and what they are used for remains open. The following section evaluates the RMF risk measurement based on a case study which contains the street sign classification on autonomous driving which is attacked by different backdoor attacks separately.

4.8. The logging function

To show all steps of the risk measurement the RMF has a function that documents them. Assigned values of the risk indicators, the pre-determined steps to measure the attacker's effort, and the calculations of the risk measurement are represented with an optimized logging function, based on the Python logging module. The function takes two arguments such as a message string and the wanted logging level (i.e. INFO or DEBUG). The following example shows an execution of the logging function:

```
1  log(f"{variable_name}", 'INFO')
```

This function should make it possible for human judgment to find the critical parts where the ML model is particularly vulnerable for backdoor attacks. The next section evaluates the implemented concept from this section and present the results of a case study which uses the RMF to measure risks of a ML model backdoor attack.

5. Evaluation

A common example to show backdoor attacks is traffic sign detection ([64], [34], [68], [51]). That makes it easier to find datasets and already finished ML models to make a case study. In the following subsections the focus lies on the case study which concept and its implementation is evaluated here.
Further, the requirements and procedures from ISO 27004 in relation with the RMF is discussed in this section. In addition, this section describes real-world examples for which the RMF could be used.

5.1. Evaluation of the ISO 27004 standard in context to the RMF

Sections 3 and 4 show both that it is possible to design and implement a technical framework for ML security especially risk measurement based on ISO 27004 [2]. Also, the decisions that come individually from the organization had to be disregarded by using specific attacks and risk indicators because there is no choice available at this point. However, as soon as individual decisions are required, such as information needs and stakeholder identification, it is no longer possible to implement this without human decisions.

5.2. Case Study: Developing a SVM for traffic sign detection

For the case study scikit-learn [72] and for preparation of the dataset in Python OpenCV2 have different function to load and resize images [17]. In their work, Stallkamp et al. [91] built a mulit-category classification dataset. The mulit-category classification dataset contains german traffic signs for image classification. That mulit-category classification dataset uses the german traffic signs from a approx. 10 hours daytime video from different roads. This case study is an example to show the functions and results of the RMF. After showing this case study there will be explain and discuss realistic case studies where backdoor attacks could have a more realistic impact for scores of ML models.

Structure of the ML model for the case study

The original dataset from Stallkamp et al. [91] is splitted between a training and testing folder. The training folder separate 43 signs into subfolders. This subfolders make it easy to use specific traffic signs which decrease the training time. The information of the folders are written in an eponymous csv-file that are not needed further in this case study. In Figure 36 the shown traffic signs can be used for training the SVM and are all labeled in the data preprocessing like the subfolder name 0 - 42. The training set contains traffic signs such as speed limit, prohibitory, derestriction, mandatory, danger, and unique signs.

All signs are resized to 50x50 pixel to optimize the performance of the NN. The training sets are also scaled with Keras and TensorFlow to execute the attacks successfully. The training data and test data load from two different folders and are trained during ten

Figure 36: Labeled traffic signs adapted from [91].

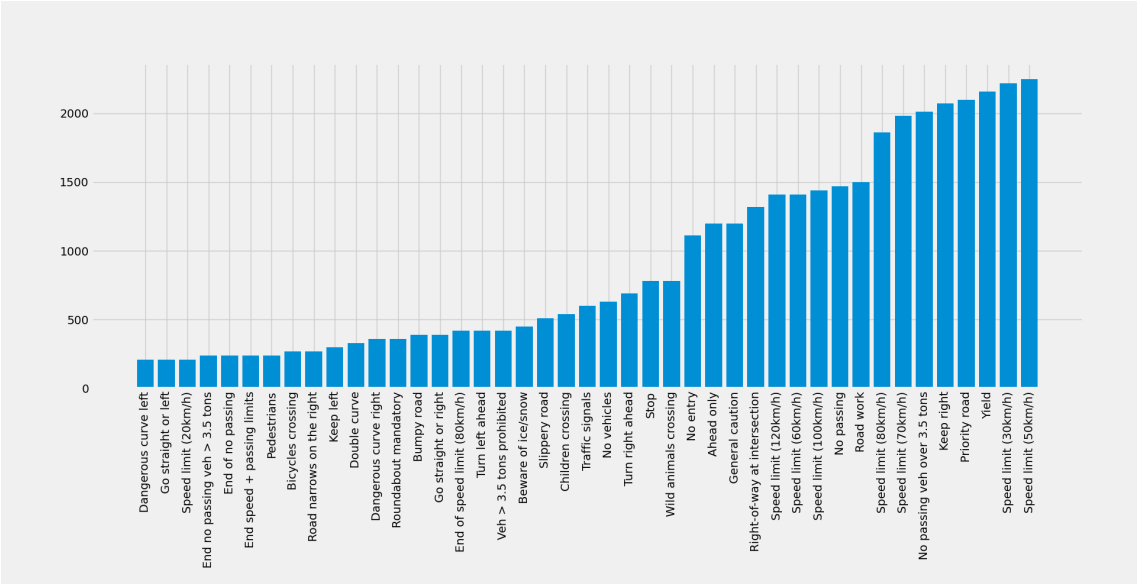

Figure 37: Number of images per labels.

epochs. All functions, the arguments and a description of them can be found in appendix B. Figure 38 shows the NN architecture.

The computer to execute the ML model have resources such as, AMD Ryzen [7] 7 5800X 8-Core Processor with 3.80 GHz, 16 GB of RAM, and a NVIDIA GeForce RTX [81] 3060 Ti. These resources are available to the attacker and to measure the computational resources to get the attacker's effort.

5.3. Differences between manipulated and original dataset

The Python plots from the case study show in this subsection examples images to show the differences between the original and manipulated dataset. Figure 39 visualizes the prediction of the ML model with the attack of the RMF. If the ML model is executed multiple times, the poisoned image show always the same wrong label because the attack specificity is targeted.

5.4. Results from the measurement methods

For this case study, the ML model is attacked by a *Clean Label Backdoor Attack* with a backdoor trigger that Figure 19 shows. 50% of the images are poisoned during the attack

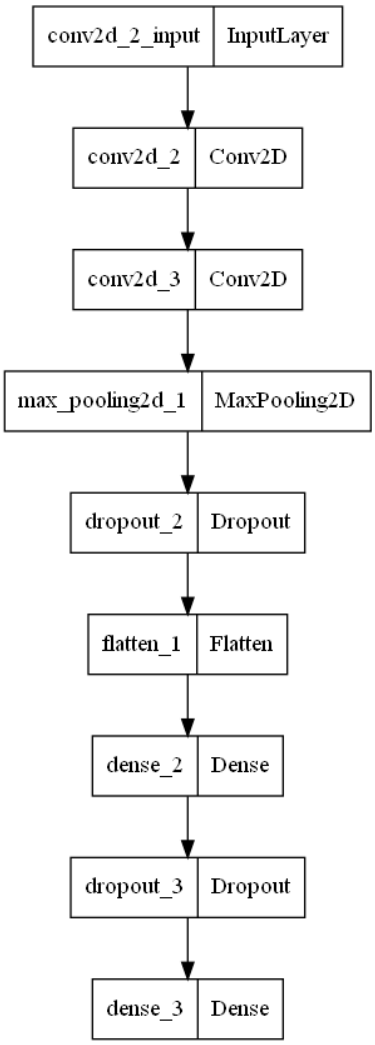

Figure 38: Generated NN architecture image from Keras.

Figure 39: The left image shows a clean output. The right image shows a poisoned output with a wrong prediction. Both images are the output from the ML model with the *Clean Label Backdoor Attack* from the RMF.

to missclassify to the label *10: No passing veh over 3.5 tons.*

Risk indicators to measure the attacker's effort

Attacker's goal As the attack is a backdoor attack, the RMF returns for the attacker's goal an inference failure. It takes six steps to develop a backdoor attack. These development steps are generalized and base not on a specific attack. The RMF returned these six steps as a natural number which shows, that the RMF measured the correct goal.

Attacker's knowledge This risk indicator returned the number of steps to implement the *Clean Label Backdoor Attack*. Based on the work of Turner et al. [94], this thesis figured out that it takes ten steps to implement this attack. The ART [66] define which information are to be used as parameters to execute this attack.

Computational resources The computational resources that are measured are the CPU, GPU, and RAM. The RAM measurement starts at the beginning of the implemented ML model and shows after finishing the training time of the ML model how much RAM resources the ML model uses. The CPU and GPU are measured directly after the training without a start point. The training time without the attack uses $970.7B$ RAM, 7.2% CPU (which is 0.07), and $8094.27MB$ of the GPU.
The training time with the implemented attack uses $1825.73MB$ RAM, 9% CPU (which is 0.09), and $8137.81MB$ of the GPU. The CPU and RAM measurement do not measure the training time of the ML model, but from the complete Python program.

Risk indicators to measure the extent of damage

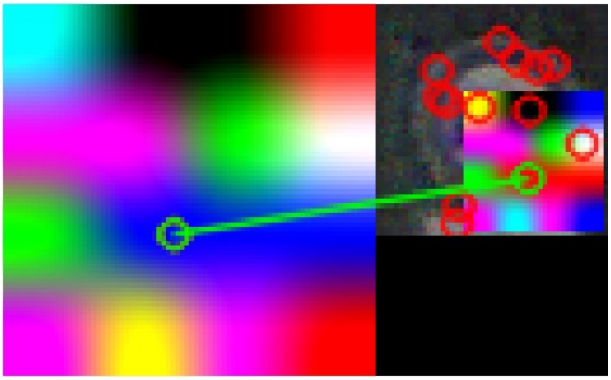

Figure 40: Backdoor image feature point matched with the backdoor in the poisoned image. To show the success of this matching, the backdoor trigger in the image on the right has been enlarged. The size of the backdoor trigger makes no difference.

Attack time All backdoor attacks are executing during training time. Figure 40 shows that FLANN and SIFT found the backdoor in a poisoned image. The time to train a

ML model without the attack and inference time takes 962.14 seconds. When the attack is added, it is 1037.23 seconds.
TP, TN, FP, FN Table 4 shows the TP, TN, FP, and FN from the last epoch before and after the attack.
Accuracy The accuracy is a risk indicator should be decreased as much as possible if an ML model is attacked with a backdoor attack. The original ML model has an accuracy of 0.94 and the accuracy with the poisoned dataset 0.06.

Attack specificity The *PoisoningAttackCleanLabelBackdoor* is a targeted attack which means for the attacker it takes four steps to choose a specific label. If there are more than one source and target label then the number of steps would be increased. The following steps must be performed:

1. Choosing the label *10: No passing veh over 3.5 tons*
2. Choose a number of images to poison
3. Choose the possible labels to poison (all labels)
4. Selecting the target images randomly to poison
5. Transmit the selected images to the PGD

5.5. Evaluation of the measurement construct

After the measurement methods, the RMF evaluates the base measures. For this purpose, all values must be unified so that they can be added together. This unification works only for the extent of damage because the corresponding risk indicators have values whose units can be adjusted in each case. The TP, TN, FP, FN is a risk indicator that calculates the ML metrics which have the same unit as the accuracy. This relation makes it possible to convert the ML metrics and the accuracy into a standartized unit. In combination with the number of successfully missclassifications which can be also despicted as the same unit as the ML metrics and the accuracy, it is possible to calculate all values with the sum formula to the extent of damage as a total value.
The attacker's effort forms from the total number of steps the attacker has to do to execute the attack, the time and the computational resources. These different risk indicators have no approach to be offset by uniform values. Therefore this does not lead to a plausible result, the values can not be combined to a common value, but can be considered individually. From this point on, the measurement construct is adjusted so that all calculated values are presented separately in different risk matrices at the end.

Derived measures

ML metrics The following Figures show the ML metrics with the original predicitions and the poisoned predicitions during inference time.

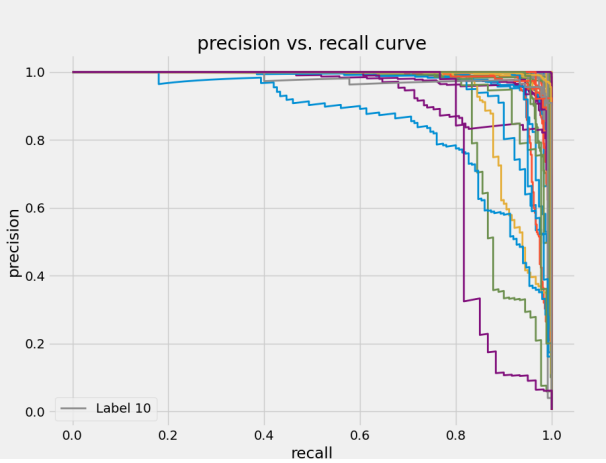

Figure 41: The precision-recall curve of the original test dataset. Each color represents a class of the ML model.

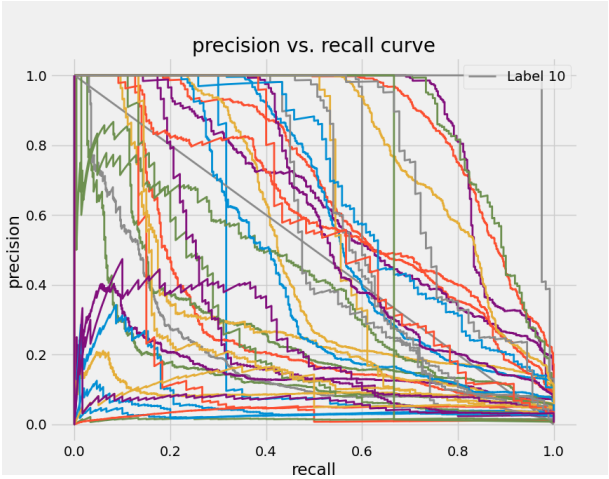

Figure 42: The precision-recall curve of the poisoned test dataset. Each color represents a class of the ML model.

Figure 41 and 42 show the different precision-recall curves of the original test dataset and the poisoned dataset. Each colored function represents the progression of a label. Figure 41 shows a clear tendency of the precision-recall curve where the recall values increase steadily while the precision values decrease that the ML model makes the best accessible distinctions. As a result, the predictions become closer to the actual results, which causes the threshold to further separate the TP and FN values from the FP and TN.
Figure 42 shows how the precision-recall curve proceeds after the original test dataset is poisoned with the same poisoning function which is used for the original training dataset. This curve shows that the clear tendency of the process for each label is no longer present except for the target label of the backdoor attack. That tendency shows the effectivness of the attack because there is only the target label which compares to the precision-recall curve of the target label.
The only label that hardly changes in Figures 41 and 42 is label *10: No passing veh over 3.5 tons*, since this was defined as the target label and is still classified as the correct label in the prediction.

Figure 43 and 44 show the different confusion matrices of the original test dataset and the poisoned dataset. The confusion matrix in Figure 43 shows the summarized predictions of the original testdata. Each column of the matrix summarizes the predictions and each row the actual labels. Most of the predictions and their actual labels run diagonally in the confusion matrix, as shown by the lighter colors. The brighter an index is, the higher its value.
In relation to the confusion matrix in Figure 44, the predictions change towards label *10: No passing veh over 3.5 tons*, since this is the target label of the attack. The matrix also

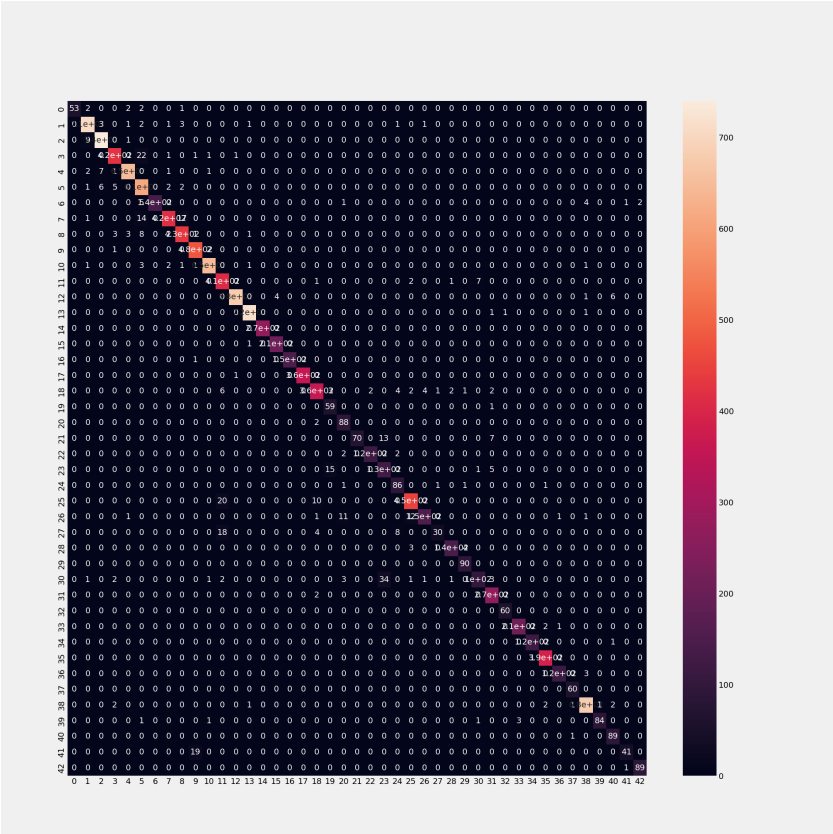

Figure 43: The confusion matrix of the original test dataset.

shows the actual labels in the darker indices. Therefore, the confusion matrix can also be used to visually depict the difference between the original and the poisoned test dataset. In order to calculate the extent of damage with the ML metrics, the precision, recall, and F1-Score are all calculated for each label and then summarized them as their average value which table 5 shows.

Derived damage values After calculating the ML metrics, this measurement function calculate based on the attack time and specificity how many images are missclassified successfully and count the number of these images. Before the counter can start, the attack time checks with the FLANN based pattern matcher if the attack worked during training time. If the attack time found images with the backdoor, then this function start to count the number of target labels. It is important that, as mentioned above, the poisoned dataset only takes images that do not already have the target label. These would otherwise falsify the result. After the comparing is finished, then this function gets the number of possible poisoned images from the attack specificity and calculates the result as the percentage value to unify the result with the other values for the extent of damage.

Derived attack steps The attack steps calculate from the attacker's knowledge, attacker's goal, and attack specificity. At this point, the attack time is no longer considered

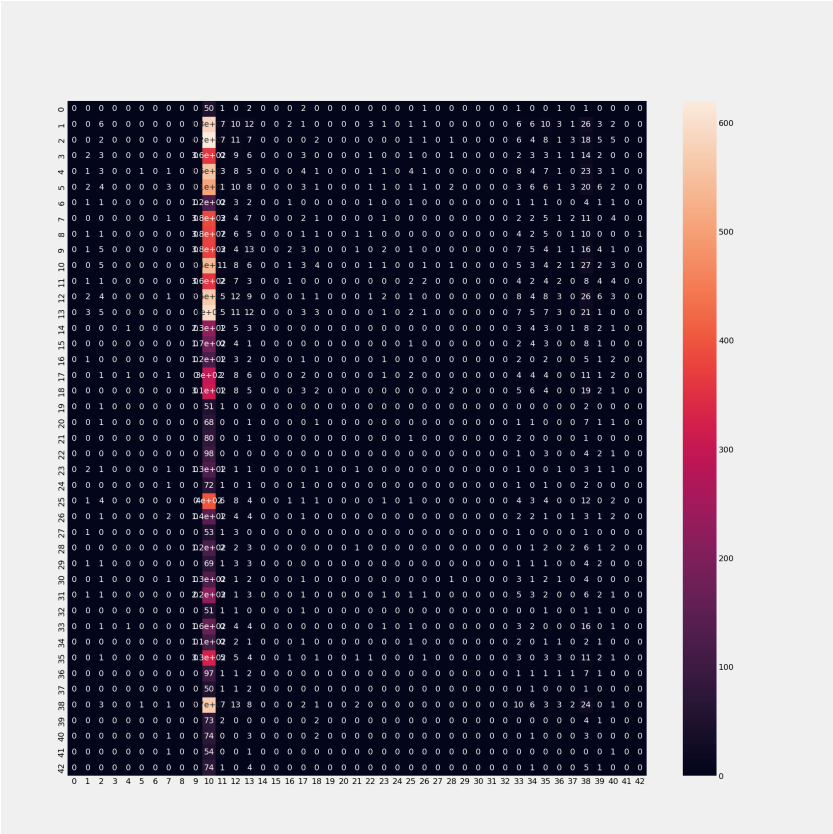

Figure 44: The confusion matrix of the poisoned test dataset.

but is passed to the analytical model as a separate value.
After calculating all steps together, the attacker has to perform 21 steps to achieve the inference failure, implement and execute the *PoisoningAttackCleanLabelBackdoor*, and poisoning a random amount of all labels to missclassify images to the target label *10: No passing veh over 3.5 tons*.
Analytical models This step in the RMF should calculate the base and derived measures originally into the extent of damage and the probability of occurrence. The extent of damage can still be calculated but the probability of occurrence cannot. The extent of damage is calculated by the reversed accuracy and average ML metrics to increase or decrease the damage the closer or further away the accuracy and average ML metrics are.

5.6. The decision criteria and measurement results

5.7. Poisoning and backdoor attacks in real applications

Beside the exemplary application from the case study, the scientific papers in this subsection show real applications where the RMF can then help in a more real environment. An example for real-world poisoning attacks against ML models is Microsoft's chatbot Tay. This chatbot learned racist and offensive language from Twitter users [12], [11].

Microsoft removed the bot after 16 hours because the bot produced offensive tweets.

6. Conclusion

This section summarizes this thesis, discusses the results related to the goals, research questions, and hypotheses from section 1. Furthermore, this section explain a possible future work based on the concept, implementation, and evaluation of the current state of the RMF.

6.1. Discussion of the results

The ISO 27004 standard is very generalized. That makes it possible to design and implement frameworks which are not covered as a standard or intended to be. The attack time is a risk indicators that measures two different outputs. This could be splitted up into two different risk indicators which would decrease the need to map it.

6.2. Optimization proposals

After implementing only backdoor attacks it should be possible in the future to use attacks for different attack times. That could increase the knowledge for vulnerabilites in the measured ML model. Further, the RMF could show how to prevent those attacks. This risk matrix is a component for which the appropriate incorporation of the risk value is still missing. Jianxing et al. [43] explain in their work that the level of risk could be suitably integrated on knowledge-based rules which this thesis did not do. This is because there is no knowledge-base which this thesis could use to integrate the risk value but could be implemented in the future. In order to increase the accuracy of the effort steps, it would be useful to add a weighting to the individual steps that describes how much effort is required for each step.
To find vulnerabilites that could execute a backdoor attack [21] describe an algorithm to identify these attacks based on activation clustering.

A. Framework functions

```
log(message, logging_levelname: str = 'INFO')
```

message output in the log

logging_levelname (Optional) string value to show the logging level

The following functions are the attacks in the RMF:

```
art_poison_backdoor_attack(x, y, num_of_rand_images)
```

x This argument takes the array of training images.

y This argument takes the array of labels that are assigned to the training images.

num_of_rand_images This argument takes the number of untargeted images that should get a trigger pattern.

```
clean_label(x, y)
```

x This argument...

y This argument...

```
art_hidden_trigger_backdoor(x, y, target, source)
```

x This argument...

y This argument...

target This argument...

source This argument...

The following functions visualize the ML model training process:

```
create_learning_curve(est, x_train, y_train, train_sz)
```

est This argument...

x_train This argument...

y_train This argument...

train_sz This argument...

```
make_meshgrid(x, y, h=0.02)
```

x Data to base x-axis meshgrid on

y Data to base y-axis meshgrid on

h (Optional) stepsize for meshgrid

```
plot_contours(ax, clf, xx, yy, **params)
```

ax Matplotlib axes object

clf Classifier

xx Meshgrid ndarray

yy Meshgrid ndarray

****params** (Optional) dictionary of params to pass to contourf

The following functions show the data results of the risk measurement:

```
probability_calculation()
```

```
list_probability()
```

The following functions represent the risk indicators in the RMF:
These three functions form the computational resources risk indicator:

```
ram_resources()
```

```
cpu_resources()
```

```
gpu_resources()
```

```
accuracy_log(true_values, predictions, normalize=False)
```

true_values

predictions

normalize=False (Optional) the accuracy can be normalized but the default value is *False*

```
attackers_knowledge(attack)
```

attack This argument takes the executed attack function to poison the training data

```
attackers_goal()
```

```
positive_negative_label(thresholds=None, name=None, dtype=None)
```

thresholds=None

name=None

dtype=None

```
attack_time(attack)
```

attack This argument takes the executed attack function to poison the training data

```
attack_specificty(target)
```

target This argument expects if the attack is targeted or untargeted

The following functions are used for the measurement construct:

```
mapping(low_l, high_l)
```

low_l This argument must be a list of all low-level attributes

high_l This argument must be a list of all high-level attributes

```
measurement_functions(base_measures)
```

base_measures This argument have to be a list of the assigend risk indicators

```
analytical_model(base_mea_raw, derived_measures)
```

base_mea_raw This argument takes a list of all base measures that are not used for the derived measures

derived_measures This argument takes a list of all derived measures from the measurement function

```
decision_criteria(*indicator)
```

***indicator** This ***args** is an argument that takes all indicators from the analytical model

B. Case Study functions

class_num This argument gets the key value from the labels which is an integer.

train_number This argument gets the number of images from a labeled folder.

train_path This argument gets the path where the local training data is stored.

data_dir This argument gets the path where all local images are stored.

image_data This argument gets an array with all images from a label.

image_labels This argument gets the label which belongs to the corresponding images.

X_train This argument gets the training data.

attack Name of a backdoor attack as a string.

```
dataset_visualization(class_num, train_number)
```

With this function, the number of images are visualized and sorted from the lowest to the highest number of images per label.

```
read_training_data(train_path, data_dir)
```

This function reads in the training data, calls the *dataset_visualization()* function and resize the images to $30x30$ pixels. The function is called by the *preprocessing()* function.

```
preprocessing(train_path, data_dir, image_data, image_labels)
```

After calling the *read_training_data()* function, this function assign the **image_date** and **image_labels** arguments to shuffle the training data. Then the training data splits into training and validation data. The shape of the images must be reshaped for the SVM.

```
model_training(train_path, data_dir, image_data, image_labels)
```

After calling the *preprocessing()* function, this function calls the *KerasClassifier()* class which is an ART class to use Keras. This class must be used because the attacks only takes these classes for the proxy model training.

```
read_test_data(train_path, data_dir, image_data, image_labels)
```

After calling the *model_training()* function, this function reads in the test data, resizes the images to $30x30$ pixels and then reshape the images. The last step is the prediction with the test data.

```
create_model(X_train)
```

This function builds the ML model architecture for the classifier class of the ART.

```
model_finetuning()
```

This function is espacially for the *Hidden Trigger Backdoor Attack* because this attack must be executed while ML model finetuning.

```
create_attack(attack)
```

To call an attack this function must be called with an attack name.

C. Risk results template

At first, the evaluated measurement shows the risk matrix.

As next, the logged values are presented to show the raw data.

After that, the ML metrics are visualized.

References

[1] Vertex ai. https://cloud.google.com/vertex-ai, accessed on 2022-03-04.

[2] *Information technoloy - Security techniques - Information security management - Measurement*. ISO, 1st edition, 2009.

[3] Cyber-glossar, Jan 2021. https://www.bsi.bund.de/DE/Service-Navi/Cyber-Glossar/cyber-glossar_node.html, accessed on 2022-24-01.

[4] Machine learning glossary, Jul 2021. https://developers.google.com/machine-learning/glossary/, accessed on 2022-24-02.

[5] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek Gordon Murray, Benoit Steiner, Paul A. Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zhang. Tensorflow: A system for large-scale machine learning. *CoRR*, abs/1605.08695, 2016.

[6] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. *CoRR*, abs/1701.07875, 2017.

[7] Sonu Arora, Dan Bouvier, and Chris Weaver. AMD next generation 7nm ryzen™ 4000 APU "renoir". In *IEEE Hot Chips 32 Symposium, HCS 2020, Palo Alto, CA, USA, August 16-18, 2020*, pages 1–30. IEEE, 2020.

[8] Iram Arshad, Mamoona Naveed Asghar, Yuansong Qiao, Brian Lee, and Yuhang Ye. Pixdoor: A pixel-space backdoor attack on deep learning models. In *29th European Signal Processing Conference, EUSIPCO 2021, Dublin, Ireland, August 23-27, 2021*, pages 681–685. IEEE, 2021.

[9] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. *ArXiv*, abs/1807.00459, 2020.

[10] Rostyslav Barabanov, Stewart Kowalski, and Louise Yngström. Information security metrics: State of the art: State of the art. 2011.

[11] Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, and Jaehoon Amir Safavi. Mitigating poisoning attacks on machine learning models: A data provenance based approach. In Bhavani M. Thuraisingham, Battista Biggio, David Mandell Freeman, Brad Miller, and Arunesh Sinha, editors, *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, AISec@CCS 2017, Dallas, TX, USA, November 3, 2017*, pages 103–110. ACM, 2017.

[12] Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, Jaehoon Amir Safavi, and Rui Zhang. Detecting poisoning attacks on machine learning in iot environments.

In *2018 IEEE International Congress on Internet of Things, ICIOT 2018, San Francisco, CA, USA, July 2-7, 2018*, pages 57–64. IEEE Computer Society, 2018.

[13] Marco Barreno, Blaine Nelson, Anthony D. Joseph, and J. D. Tygar. The security of machine learning. *Mach. Learn.*, 81(2):121–148, 2010.

[14] Marco Barreno, Blaine Nelson, Russell Sears, Anthony D. Joseph, and J. D. Tygar. Can machine learning be secure? In Ferng-Ching Lin, Der-Tsai Lee, Bao-Shuh Paul Lin, Shiuhpyng Shieh, and Sushil Jajodia, editors, *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security, ASIACCS 2006, Taipei, Taiwan, March 21-24, 2006*, pages 16–25. ACM, 2006.

[15] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, 1975.

[16] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress, 2012.

[17] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[18] Jakub Breier, Adrian Baldwin, Helen Balinsky, and Yang Liu. Risk management framework for machine learning security. *CoRR*, abs/2012.04884, 2020.

[19] Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pages 39–57. IEEE Computer Society, 2017.

[20] Nicholas Carlini and David A. Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. In *2018 IEEE Security and Privacy Workshops, SP Workshops 2018, San Francisco, CA, USA, May 24, 2018*, pages 1–7. IEEE Computer Society, 2018.

[21] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian M. Molloy, and Biplav Srivastava. Detecting backdoor attacks on deep neural networks by activation clustering. *CoRR*, abs/1811.03728, 2018.

[22] Tong Chen and Zhan Ma. Towards robust neural image compression: Adversarial attack and model finetuning. *CoRR*, abs/2112.08691, 2021.

[23] Jesse Davis and Mark Goadrich. The relationship between precision-recall and ROC curves. In William W. Cohen and Andrew W. Moore, editors, *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, volume 148 of *ACM International Conference Proceeding Series*, pages 233–240. ACM, 2006.

[24] Elena Doynikova, Evgenia Novikova, Diana Gaifulina, and Igor V. Kotenko. Towards attacker attribution for risk analysis. In Joaquín García-Alfaro, Jean Leneutre, Nora Cuppens, and Reda Yaich, editors, *Risks and Security of Internet and Systems - 15th International Conference, CRiSIS 2020, Paris, France, November 4-6, 2020, Revised Selected Papers*, volume 12528 of *Lecture Notes in Computer Science*, pages 347–353. Springer, 2020.

[25] Eduardo Fonseca, Manoj Plakal, Frederic Font, Daniel P. W. Ellis, and Xavier Serra. Audio tagging with noisy labels and minimal supervision. In Michael I. Mandel, Justin Salamon, and Daniel P. W. Ellis, editors, *Proceedings of the Workshop on Detection and Classification of Acoustic Scenes and Events 2019 (DCASE 2019), New York University, NY, USA, October 2019*, pages 69–73, 2019.

[26] Chollet François. *Deep learning mit Python und Keras das Praxis-Handbuch vom Entwickler der Keras-Bibliothek*. mitp, 2018.

[27] Daniel Fraunholz, Daniel Krohmer, Simon Duque Antón, and Hans Dieter Schotten. YAAS - on the attribution of honeypot data. *Int. J. Cyber Situational Aware.*, 2(1):31–48, 2017.

[28] Cong Fu and Deng Cai. EFANNA : An extremely fast approximate nearest neighbor search algorithm based on knn graph. *CoRR*, abs/1609.07228, 2016.

[29] Bundesamt für Sicherheit in der Informationstechnik (BSI). *Application of Attack Potential to Smartcards*. 2013. `https://www.commoncriteriaportal.org/files/supdocs/CCDB-2013-05-002.pdf`, accessed on 2022-15-04.

[30] Salvador García, Alberto Fernández, Julián Luengo, and Francisco Herrera. A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability. *Soft Comput.*, 13(10):959–977, 2009.

[31] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014.

[32] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[33] Daniele Grattarola and Cesare Alippi. Graph neural networks in tensorflow and keras with spektral. *CoRR*, abs/2006.12138, 2020.

[34] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *CoRR*, abs/1708.06733, 2017.

[35] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5767–5777, 2017.

[36] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017.

[37] Kenta Hara and Yumi Asahi. Factor analysis of continuous use of car services in japan by machine learning. In Sakae Yamamoto and Hirohiko Mori, editors, *Human Interface and the Management of Information. Information-Rich and Intelligent Environments - Thematic Area, HIMI 2021, Held as Part of the 23rd HCI International Conference, HCII 2021, Virtual Event, July 24-29, 2021, Proceedings, Part II*, volume 12766 of *Lecture Notes in Computer Science*, pages 373–384. Springer, 2021.

[38] Jamie Hayes and Olga Ohrimenko. Contamination attacks and mitigation in multi-party machine learning. In *NeurIPS*, 2018.

[39] Hailong Hu and Jun Pang. Stealing machine learning models: Attacks and countermeasures for generative adversarial networks. In *ACSAC '21: Annual Computer Security Applications Conference, Virtual Event, USA, December 6 - 10, 2021*, pages 1–16. ACM, 2021.

[40] Ling Huang, Anthony D. Joseph, Blaine Nelson, Benjamin I. P. Rubinstein, and J. D. Tygar. Adversarial machine learning. In Yan Chen, Alvaro A. Cárdenas, Rachel Greenstadt, and Benjamin I. P. Rubinstein, editors, *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence, AISec 2011, Chicago, IL, USA, October 21, 2011*, pages 43–58. ACM, 2011.

[41] Olena I. Ivanenko. Implementation of risk assessment for critical infrastructure protection with the use of risk matrix. 2020.

[42] Marko Jahnke, Christian Thul, and Peter Martini. Graph based metrics for intrusion response measures in computer networks. In *32nd Annual IEEE Conference on Local Computer Networks (LCN 2007), 15-18 October 2007, Clontarf Castle, Dublin, Ireland, Proceedings*, pages 1035–1042. IEEE Computer Society, 2007.

[43] Yu Jianxing, Chen Haicheng, Wu Shibo, and Fan Haizhao. A novel risk matrix approach based on cloud model for risk assessment under uncertainty. *IEEE Access*, 9:27884–27896, 2021.

[44] Byunggill Joe, Akshay Mehra, Insik Shin, and Jihun Hamm. Machine learning with electronic health records is vulnerable to backdoor trigger attacks. *CoRR*, abs/2106.07925, 2021.

[45] Heinrich Kersten, Jürgen Reuter, Klaus-Werner Schröder, and Klaus-Dieter Wolfenstetter. *IT-Sicherheitsmanagement nach ISO 27001 und Grundschutz*. Springer Vieweg, 4th edition, 2013.

[46] Nizar Kheir, Nora Cuppens-Boulahia, Frédéric Cuppens, and Hervé Debar. A service dependency model for cost-sensitive intrusion response. In Dimitris Gritzalis, Bart Preneel, and Marianthi Theoharidou, editors, *Computer Security - ESORICS 2010, 15th European Symposium on Research in Computer Security, Athens, Greece, September 20-22, 2010. Proceedings*, volume 6345 of *Lecture Notes in Computer Science*, pages 626–642. Springer, 2010.

[47] Yoon Kim. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882, 2014.

[48] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[49] Yann LeCun, Corinna Cortes, and Christopher J.C. Burges. The mnist database, 2017.

[50] Yann LeCun, Lawrence D. Jackel, Léon Bottou, Corinna Cortes, John S. Denker, Harris Drucker, Isabelle Guyon, Urs Muller, E. Sackinger, Patrice Y. Simard, and Vladimir Naumovich Vapnik. Learning algorithms for classification: A comparison on handwritten digit recognition. 1995.

[51] Shaofeng Li, Minhui Xue, Benjamin Zi Hao Zhao, Haojin Zhu, and Xinpeng Zhang. Invisible backdoor attacks on deep neural networks via steganography and regularization. *IEEE Trans. Dependable Secur. Comput.*, 18(5):2088–2105, 2021.

[52] Xian Li, Changhan Wang, Yun Tang, Chau Tran, Yuqing Tang, Juan Miguel Pino, Alexei Baevski, Alexis Conneau, and Michael Auli. Multilingual speech translation from efficient finetuning of pretrained models. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 827–838. Association for Computational Linguistics, 2021.

[53] Jing Lin, Long Dang, Mohamed Rahouti, and Kaiqi Xiong. ML attack models: Adversarial attacks and data poisoning attacks. *CoRR*, abs/2112.02797, 2021.

[54] Kristoffer Lundholm, Jonas Hallberg, and Helena Granlund. Design and use of information security metrics. *FOI, Swedish Def. Res. Agency, p. ISSN*, pages 1650–1942, 2011.

[55] Yuxin Ma, Tiankai Xie, Jundong Li, and Ross Maciejewski. Explaining vulnerabilities to adversarial machine learning through visual analytics. *CoRR*, abs/1907.07296, 2019.

[56] Andrew L. Maas. Rectifier nonlinearities improve neural network acoustic models. 2013.

[57] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *CoRR*, abs/1706.06083, 2017.

[58] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[59] Diganta Misra. Mish: A self regularized non-monotonic activation function. In *31st British Machine Vision Conference 2020, BMVC 2020, Virtual Event, UK, September 7-10, 2020*. BMVA Press, 2020.

[60] Jojo Moolayil. *Learn keras for deep neural networks: A fast-track approach to modern deep learning with python*. Apress, 2019.

[61] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 86–94. IEEE Computer Society, 2017.

[62] Marius Muja and David G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(11):2227–2240, 2014.

[63] Chi Nhan Nguyen and Oliver Zeigermann. *Machine Learning kurz & gut*. O'Reilly Verlag, 2018.

[64] Tuan Anh Nguyen and Anh Tuan Tran. Wanet - imperceptible warping-based backdoor attack. *CoRR*, abs/2102.10369, 2021.

[65] Maria-Irina Nicolae, Mathieu Sinn, Tran Ngoc Minh, Ambrish Rawat, Martin Wistuba, Valentina Zantedeschi, Ian M. Molloy, and Benjamin Edwards. Adversarial robustness toolbox v1.0.0. *CoRR*, abs/1807.01069, 2019.

[66] Maria-Irina Nicolae, Mathieu Sinn, Minh Ngoc Tran, Beat Buesser, Ambrish Rawat, Martin Wistuba, Valentina Zantedeschi, Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, Ian Molloy, and Ben Edwards. Adversarial robustness toolbox v1.2.0. *CoRR*, 1807.01069, 2018.

[67] Bojan Novkovic. A taxonomy of defenses against memory corruption attacks. In Marko Koricic, Karolj Skala, Zeljka Car, Marina Cicin-Sain, Snjezana Babic, Vlado Sruk, Dejan Skvorc, Slobodan Ribaric, Bojan Jerbic, Stjepan Gros, Boris Vrdoljak, Mladen Mauher, Edvard Tijan, Tihomir Katulic, Juraj Petrovic, Tihana Galinac Grbac, Nikola Filip Fijan, and Vera Gradisnik, editors, *44th International Convention on Information, Communication and Electronic Technology, MIPRO 2021, Opatija, Croatia, September 27 - Oct. 1, 2021*, pages 1196–1201. IEEE, 2021.

[68] Florian Nuding and Rudolf Mayer. Poisoning attacks in federated learning: An evaluation on traffic sign classification. In Vassil Roussev, Bhavani M. Thuraisingham, Barbara Carminati, and Murat Kantarcioglu, editors, *CODASPY '20: Tenth ACM Conference on Data and Application Security and Privacy, New Orleans, LA, USA, March 16-18, 2020*, pages 168–170. ACM, 2020.

[69] National Institute of Standards and Technology. Security requirements for cryptographic modules. Technical Report Federal Information Processing Standards Publications (FIPS PUBS) 140-2, Change Notice 2 December 03, 2002, U.S. Department of Commerce, Washington, D.C., 2001.

[70] Markos Papadonikolakis, Christos-Savvas Bouganis, and George A. Constantinides. Performance comparison of gpu and fpga architectures for the svm training problem. *2009 International Conference on Field-Programmable Technology*, pages 388–391, 2009.

[71] Nicolas Papernot, Patrick D. McDaniel, Ian J. Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In Ramesh Karri, Ozgur Sinanoglu, Ahmad-Reza Sadeghi, and Xun Yi, editors, *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017, Abu Dhabi, United Arab Emirates, April 2-6, 2017*, pages 506–519. ACM, 2017.

[72] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[73] Binh Thai Pham, Abolfazl Jaafari, Mohammadtaghi Avand, Nadhir Al-Ansari, Tran Dinh Du, Hoang Phan Hai Yen, Tran Van Phong, Duy Huu Nguyen, Hiep Van Le, Davood Mafi Gholami, Indra Prakash, Hoang Thi Thuy, and Tran Thi Tuyen. Performance evaluation of machine learning methods for forest fire modeling and prediction. *Symmetry*, 12(6):1022, 2020.

[74] Katharina Prinz and Arthur Flexer. End-to-end adversarial white box attacks on music instrument classification. *CoRR*, abs/2007.14714, 2020.

[75] Daniel S. Quintana. A synthetic dataset primer for the biobehavioural sciences to promote reproducibility and hypothesis generation. *eLife*, 9, 2020.

[76] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 91–99, 2015.

[77] Ishai Rosenberg, Asaf Shabtai, Yuval Elovici, and Lior Rokach. Adversarial machine learning attacks and defense methods in the cyber security domain. *ACM Comput. Surv.*, 54(5):108:1–108:36, 2021.

[78] Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J. Fleet. Adversarial manipulation of deep representations. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[79] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. Hidden trigger backdoor attacks. *CoRR*, abs/1910.00033, 2019.

[80] Ahmed Salem, Rui Wen, Michael Backes, Shiqing Ma, and Yang Zhang. Dynamic backdoor attacks against machine learning models. *CoRR*, abs/2003.03675, 2020.

[81] Vadim V. Sanzharov, Vladimir A. Frolov, and Vladimir A. Galaktionov. Survey of nvidia RTX technology. *Program. Comput. Softw.*, 46(4):297–304, 2020.

[82] Benjamin Scherbaum, Jan 2022. `https://cybersecurity-navigator.de/norm/293`, accessed on 2022-05-19.

[83] Avi Schwarzschild, Micah Goldblum, Arjun Gupta, John P. Dickerson, and Tom Goldstein. Just how toxic is data poisoning? A unified benchmark for backdoor and data poisoning attacks. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 9389–9398. PMLR, 2021.

[84] Paul Schwerdtner, Florens Greßner, Nikhil Kapoor, Felix Assion, René Sass, Wiebke Günther, Fabian Hüger, and Peter Schlicht. Risk assessment for machine learning models. *CoRR*, abs/2011.04328, 2020.

[85] Alireza Shameli Sendi, Rouzbeh Aghababaei-Barzegar, and Mohamed Cheriet. Taxonomy of information security risk assessment (ISRA). *Comput. Secur.*, 57:14–30, 2016.

[86] Ali Shafahi, W. Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. *CoRR*, abs/1804.00792, 2018.

[87] Helen Sharp, Anthony Finkelstein, and Galal Galal. Stakeholder identification in the requirements engineering process. In *10th International Workshop on Database & Expert Systems Applications, Florence, Italy, September 1-3, 1999, Proceedings*, pages 387–391. IEEE Computer Society, 1999.

[88] Robert W. Shirey. Internet security glossary, version 2. *RFC*, 4949:1–365, 2007.

[89] Adam Shostack. *Threat Modeling : Designing for Security.* John Wiley & Sons, Incorporated, 1st edition, 2017.

[90] Zhang Songtao, Li Chao, and Lin Liqing. An improved method for eliminating false matches. *2017 2nd International Conference on Image, Vision and Computing (ICIVC)*, pages 133–137, 2017.

[91] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: A multi-class classification competition. In *The 2011 International Joint Conference on Neural Networks, IJCNN 2011, San Jose, California, USA, July 31 - August 5, 2011*, pages 1453–1460. IEEE, 2011.

[92] Elham Tabassi, Kevin Burns, Michael Hadjimichael, Andres Molina-Markham, and Julian Sexton. A taxonomy and terminology of adversarial machine learning. *NIST IR*, 2019.

[93] Marte Tarnes. Information security metrics: An empirical study of current practice. *Specialization Project, Trondheim, 17th December*, 2012.

[94] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Clean-label backdoor attacks. 2018.

[95] Sakshi Udeshi, Shanshan Peng, Gerald Woo, Lionell Loh, Louth Rawshan, and Sudipta Chattopadhyay. Model agnostic defence against backdoor attacks in machine learning. *IEEE Trans. Reliab.*, 71(2):880–895, 2022.

[96] Ivan Vaccari, Maurizio Aiello, and Enrico Cambiaso. Slowite, a novel denial of service attack affecting MQTT. *Sensors*, 20(10):2932, 2020.

[97] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual.* CreateSpace, Scotts Valley, CA, 2009.

[98] Jia Wan, Ziquan Liu, and Antoni B. Chan. A generalized loss function for crowd counting and localization. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 1974–1983. Computer Vision Foundation / IEEE, 2021.

[99] Yixu Wang, Jie Li, Hong Liu, Yongjian Wu, and Rongrong Ji. Black-box dissector: Towards erasing-based hard-label model stealing attack. *CoRR*, abs/2105.00623, 2021.

[100] Qixue Xiao, Kang Li, Deyue Zhang, and Weilin Xu. Security risks in deep learning implementations. In *2018 IEEE Security and Privacy Workshops, SP Workshops 2018, San Francisco, CA, USA, May 24, 2018*, pages 123–128. IEEE Computer Society, 2018.

[101] Jianfeng Xu, Yuanjian Zhang, and Duoqian Miao. Three-way confusion matrix for classification: A measure driven view. *Inf. Sci.*, 507:772–794, 2020.

[102] Mingfu Xue, Chongyan Gu, Weiqiang Liu, Shichao Yu, and Máire O'Neill. Ten years of hardware trojans: a survey from the attacker's perspective. *IET Comput. Digit. Tech.*, 14(6):231–246, 2020.

[103] Hongpo Zhang, Ning Cheng, Yang Zhang, and Zhanbo Li. Label flipping attacks against naive bayes on spam filtering systems. *Appl. Intell.*, 51(7):4503–4514, 2021.

[104] Liqiang Zhang, Guanjun Lin, Bixuan Gao, Zhibao Qin, Yonghang Tai, and Jun Zhang. Neural model stealing attack to smart mobile device on intelligent medical platform. *Wirel. Commun. Mob. Comput.*, 2020:8859489:1–8859489:10, 2020.

[105] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. *CoRR*, abs/1707.01083, 2017.

[106] Liuwan Zhu, Rui Ning, Chunsheng Xin, Chonggang Wang, and Hongyi Wu. CLEAR: clean-up sample-targeted backdoor in neural networks. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 16433–16442. IEEE, 2021.

Label	TP	TN	FP	FN
Speed limit (20km/h)	59	12570	0	1
Speed limit (30km/h)	707	11898	12	13
Speed limit (50km/h)	743	11873	7	7
Speed limit (60km/h)	430	12169	11	20
Speed limit (70km/h)	655	11959	11	5
Speed limit (80km/h)	620	11953	47	10
End of speed limit (80km/h)	132	12480	0	18
Speed limit (100km/h)	428	12175	5	22
Speed limit (120km/h)	431	12173	7	19
No passing	479	12132	18	1
No passing veh over 3.5 tons	656	11968	2	4
Right-of-way at intersection	416	12204	6	4
Priority road	682	11937	3	8
Yield	718	11907	3	2
Stop	269	12360	0	1
No vehicles	210	12413	7	0
Veh > 3.5 tons prohibited	149	12478	2	1
No entry	351	12269	1	9
General caution	383	12229	11	7
Dangerous curve left	60	12562	8	0
Dangerous curve right	89	12522	18	1
Double curve	84	12538	2	6
Bumpy road	109	12509	1	11
Slippery road	144	12477	3	6
Road narrows on the right	89	12538	2	1
Road work	476	12128	22	4
Traffic signals	170	12445	5	10
Pedestrians	60	12568	2	0
Children crossing	147	12478	2	3
Bicycles crossing	87	12540	0	3
Beware of ice/snow	133	12480	0	17
Wild animals crossing	266	12353	7	4
End speed + passing limits	60	12568	2	0
Turn right ahead	209	12417	3	1
Turn left ahead	119	12508	2	1
Ahead only	389	12236	4	1
Go straight or right	115	12508	2	5
Go straight or left	58	12569	1	2
Keep right	684	11934	6	6
Keep left	88	12540	0	2
Roundabout mandatory	87	12535	5	3
End of no passing	45	12561	9	15
End no passing veh > 3.5 tons	79	12534	6	11

Table 4: True and False values (left) before the attack, (right) after the attack.

ML metric	Original value	Poisoned value
Average Precision	0.96	0.02
Average Recall	0.94	0.02
Average F1-Score	0.94	0.01

Table 5: Average ML metrics value between the original and the poisoned test dataset.

Selbständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und noch nicht für andere Prüfungen eingereicht habe. Sämtliche Quellen einschließlich Internetquellen, die unverändert oder abgewandelt wiedergegeben werden, insbesondere Quellen für Texte, Grafiken, Tabellen und Bilder, sind als solche kenntlich gemacht. Mir ist bekannt, dass bei Verstößen gegen diese Grundsätze ein Verfahren wegen Täuschungsversuchs bzw. Täuschung eingeleitet wird.

Berlin, den 10. Juli 2022 .....