SPARTA

D7.3

Validation and evaluation plan

Project number	830892
Project acronym	SPARTA
Project title	Strategic programs for advanced research and technology in Europe
Start date of the project	1st February, 2019
Duration	36 months
Programme	H2020-SU-ICT-2018-2020

Deliverable type	Report
Deliverable reference number	SU-ICT-03-830892 / D7.3 / V1.1
Work package contributing to the deliverable	WP7
Due date	October 2020 – M21
Actual submission date	11th June, 2021

Responsible organisation	TUM
Editor	Mohammad Reza Norouzian
Dissemination level	PU
Revision	V1.1

Abstract	This report specifies the initial plan, data and procedures for testing and evaluation of SPARTA SAFAIR solutions. The focus is on protection against adversarial attacks and ensuring machine learning models' robustness. The report describes an open contest that will be organized by SAFAIR program to check the proposed solutions. Besides, we will develop a machine learning adversarial tool to benchmark machine learning solutions in a standardised way.
Keywords	Adversarial Machine Learning, Secure AI, Robustness, Testing, Validation, AI Contest

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 830892.

Editor

Mohammad Reza Norouzian (TUM)

Contributors (ordered according to beneficiary numbers)

Zakaria Chihani (CEA)

Erkuden Rios, Eider Iturbe, Maria Carmen Palacios, Cristina Martinez (TEC)

Raúl Orduña, Xabier Etxeberria, Amaia Gil (VICOM)

Boussad Addad (TCS)

Marek Pawlicki, Michal Choras (ITTI)

Reviewers (ordered according to beneficiary numbers)

Hervé Debar (IMT)

Rimantas Zylius (L3CE)

Executive Summary

Machine Learning (ML) algorithms have been increasingly adopted in the security application domain for tasks like spam, intrusion, malware detection, and biometric authentications (e.g., facial recognition system). This is owing to the ability of these models to perform well on novel attacks and variations of the ones the model was trained on. However, as these models are being used in actual production code, the underlying assumption of data stationarity, i.e., the train and test data is tampered. The situation gets worse if we have malicious adversaries trying to take advantage of the system. For instance, the adversary may purposely manipulate data to compromise learning or inference. Such attacks initiate the concern for evaluation of ML algorithms security that is driving the research area of adversarial machine learning [1].

Due to the needs for proposing and implementing new methods to evaluate the adversarial machine learning solutions, this document proposes the initial plan and procedure to test and evaluate the WP7 SAFAIR (Secure and Fair AI Systems for Citizens) program proposed solutions adversarial machine learning. This document is the result of three months of work for planning to evaluate the solutions of SAFAIR, and the results of this evaluation will be published in D7.6 (validation and evaluation report) deliverable in M36.

To this end, we propose two approaches to address the testing and evaluation of adversarial ML and robustness of ML methods. In the first approach, we will design an open AI contest. In the contest, we will pit models against various adversarial perturbations. This shall facilitate measurable progress towards robust machine learning models. The submitting models and attacks will continuously pit against each other on an image classification task, and at the end of the contest, we will publish the leaderboard (participants ranking) and their open source solutions.

In the second approach, we plan to implement an open-source python tool. The tool will provide standardised benchmark on the performances of models in adversarial machine learning domain. It will also ease in evaluating performance of the model in adversarial setting.

It is important to note that the focus of this document is on adversarial machine learning, not Artificial Intelligence (AI) systems in general (such as expert systems, reasoners, fuzzy systems, etc.). Nonetheless, because this delivery is the scope of the SAFAIR program tasks, we will refer to ML concepts as AI concepts in this document.

Table of Content

Chapter 1 Introduction

Recent advances in machine learning and deep neural networks have enabled the application of such models to solve various problems in the domains of image processing, text classification, speech, and others. However, although these models have a very impressive performance for unperturbed samples, most existing machine learning-based models are highly vulnerable to adversarial examples which SAFAIR program was delivered an extensive report in D7.1 a knowledge base threat modeling for AI systems in a comprehensive description and documented examples of techniques to implement such attacks.

An adversarial example is generated by perturbing an input sample by a small amount but in a very specific way. Such perturbations are imperceptible to humans but can easily cause the machine learning models to produce incorrect output, for instance, a classifier to misclassify the sample. Presence of such adversarial examples which are imperceptible to humans but can fool the machine learning model, pose a serious security threat. An adversary can manage to fool production deployed systems by building surrogate models in case they do not have access to underlying models. As such, even limited access to the model is sufficient to cause havoc. Moreover, since the models would be interacting with real-world data by means of their sensor inputs, the adversary has ample opportunity to inject adversarial inputs into the model. Going by recent trends, we expect AI and machine learning models to get increasingly deployed for real-world tasks. Hence, machine learning security vulnerabilities such as adversarial examples could be used to compromise such systems. Therefore, in AI safety, robustness to adversarial examples is a crucial element of machine learning [2].

Evaluation of adversarial attacks and defences is not trivial. For the evaluation of proposed attacks or proposed defences, we can not limit ourselves to a few well-known methods for generating adversarial or defending against them. Traditional machine learning methods assume that the test data has a similar distribution to the data it was trained on. However, in an Adversarial machine learning setting, one must evaluate the techniques in an open-ended scenario. Since the models oftentimes interact with their sensor inputs or raw user data, it is possible for the model to receive inputs from an unknown distribution. Thereby, benchmarking the defence technique against a single attack or even a suite of well-known attacks before production deployment is insufficient. Even if the defence was robust in its pre-deployment evaluation, it can easily be fooled by a new attack that the defensive mechanisms did not anticipate [2]. Ideally, we would like to have some theoretical guarantees about the robustness of the models. However, such guarantees become quickly intractable for machine learning models and the problem is even worse for the deep learning models.

We propose a dual approach to tackle these issues. First, we propose a contest of solutions against adversarial machine learning attacks, which gives a useful intermediate form of evaluation. Each defence is pitted against attacks built by the participating teams. The aim of the attacking team is to fool the model by means of employing adversarial perturbations while the defence team tries to be robust to such perturbations. The evaluation of such scenarios are not as conclusive as theoretical proof but, however, it represents a better real-word case studies [2]. The evaluation carried out by the proposer of a defence technique though valid has no guarantee about the techniques it was not evaluated against. As such, having an open-ended evaluation scheme is much more beneficial.

Second, we propose a method and tool that supports developing more robust ML models. It will provide standardised benchmark on the performances of models. The proposed tool will provide reference implementations of the attacks, which are intended to be used for constructing more robust models and motivate researchers and developers to use the standardised reference implementation of attack and defences.

Chapter 2 Adversarial examples

Adversarial examples are inputs to a machine learning model which are specifically designed to fool the system and cause a misclassification. The term “clean example” is used for an input, if it is naturally occurring. This enables us to define an adversarial example as one which has been modified by an adversary such that it is misclassified. Such adversarially created examples are not guaranteed to succeed though and it is still possible that the model can classify the input correctly. As such, it is possible to measure the accuracy or error rate of a model on a particular set of adversarial examples.

2.1 Well-known adversarial attack scenarios

The attacks methods can be broadly classified as:

- **Targeted attack**. In such a scenario, the attacker tries to modify the classifier outputs in order to predict some particular class label.
- **Non-targeted attack**. In such a scenario, the attacker tries to modify the classifier outputs in order to predict any incorrect class label.

2.2 Attack methods

The attacks discussed here are the ones described in the SPARTA deliverable D7.1 and tested in the SPARTA deliverable D7.2:

Fast Gradient Sign Method (FGSM). FGSM attack is one of the widest used algorithms to produce adversarial examples [4]. The method works by using the gradients of the loss by including the input data to find a noise vector which points in the opposite direction of the loss function and maximizes the loss in that way. A non-targeted arbitrary adversarial example can be created by adding noise to a given input data sample. Since adversary is interested in creating an input dataset which maximizes the loss, the method searches for the value change in each dimension to do so. The term "fast" in the name origins from the fact that one can compute the corresponding changes very fast by applying the chain rule to the ML model, which is also used to train it. The FGSM method can also be used to generate targeted adversarial examples.

Basic Iterative Method (BIM). After further development of FGSM attack the basic iterative method was introduced [5] by scaling the noise vector down and applying the FGSM multiple times. However, adversary can maximize the loss by reaching the border of the ε norm hyper ball.

Projected Gradient Descent (PGD). In contrast to the normal gradient descent method, the projected gradient descent minimizes the gradient function in subject to a constraint. This constraint could be the false label for a given dataset. The PGD attack [6] introduces a method which is not altering the input data to a given label but can train the noise from ground up towards a given false label.

Carlini and Wagner attack (CW). N. Carlini and D. Wagner [7] introduced a method to handles the adversarial attack as a minimization problem which alters the input data in such a way that the loss to a given target label is minimized.

2.3 Overview of defences (countermeasures)

Defending methods against adversarial examples are still not satisfying, and it requires more research in most cases. We give an overview of the defences proposed so far in previous deliverable

D7.2 where SAFAIR program proposed and implemented the following defensive mechanisms and tools in D7.2:

- Adversarial (re)Training
- Dimensionality Reduction
- Prediction Similarity
- Model Behaviour
- Neural Activation-based Adversarial Attack detector

These defences were implemented in the following domains:

- Medical images
- In PDFs
- In Cybersecurity (network intrusion detection)

Adversarial Training uses the retraining as a defence mechanism. This retrains the original model by adding known adversarial examples to the training data. The purpose of this technique is that the model learns to classify them correctly, rendering the known adversarial examples ineffective. However, it has not achieved competitive robustness against new adversarial examples [8].

The **Dimensionality Reduction** defences can be implemented in several ways with different effectiveness in strengthening the model. However, all the variants follow the same idea: passing data through a dimensionality reduction layer (autoencoder and encoder layers, in our case) to remove as much noise as possible from the input image. Thus, the model can generalize, avoiding adversarial examples [9].

The **Prediction Similarity** studies the behaviour of a model according to the received inputs. The idea of this defence is based on the need for multiple predictions of similar images in order to find an effective adversarial image. For that purpose, this defence adds an external layer to the original model, which saves the history of input images used for prediction and other parameters of the predict action (such as, user, prediction values and distance to previous images) [10].

In **Model's Behaviour,** an adversarial detector is generated behind the idea that the model's behaviour changes depending on if the input is an original or adversarial image. For model's behaviour characterization aggregate functions are applied to model's layers. The aggregate functions selected are used to obtain a statistical description of the layer activation values by means of features indicating the centrality, asymmetry and dispersion of the distribution [11].

The **Neural Activation-based Adversarial Attack detector** was tested in the network intrusion detection context. The method uses the values of neural activations collected from an operational neural network trained on cybersecurity data to create a separate machine learning model capable of detecting adversarial attacks. The details of this approach, which was developed fully in SPARTA, can be found in [12].

Chapter 3 The SAFAIR AI contest

The form of the publicly open contest has shown itself as one of the most powerful and successful ways to force and accelerate the research in various areas of machine learning in recent years [13] [2] [14]. The idea of a competition in adversarial machine learning is especially promising because participants will have a choice. As adversaries to the system, they can attempt to perform attacks and try to tamper the system, or, as defenders of the system, they can attempt to build robust models and prepare the system to endure adversarial examples attacks. This is the key advantage since it is always extremely hard to benchmark and verify the robustness of a machine learning model: we cannot predict all possible attacks and defences. Additionally, the SAFAIR AI contest simulates well the real-life IT security scenarios, in which one party is always trying to find new ways to cheat, and the other is constantly devising appropriate defences.

3.1 Objectives

Adversarial perturbations show that decision-making in modern deep neural networks is driven by correlation rather than causation features. It's a crucial concern from a security viewpoint because adversarial perturbations can severely affect machine decisions without noticing humans [14].

However, the problem for precise evaluation of model robustness is remaining. It would be a mistake to assume that the model is robust because it was robust to a particular attack or an attack-suite. Methods such as gradient masking are often employed to handle various adversarial attacks. However, recent researches show that it is possible to easily by-pass such methods and still cause misclassifications. Therefore, similar to cryptography, testing model robustness is particularly bound to how well the model is standing against the specific attacks they are designed for.

The SAFAIR AI contest is designed as a two-player game. Here a machine learning models (based on some defence technique) is pitted against attacks designed to cause misclassification. This enables an understanding of the effectiveness of the various attack and defence methods by means of a continuous evaluation. Since the given attack tries to break multiple defence techniques and similarly, a defence tries to evaluate its effectiveness against different attack techniques, this provides us with a better sense of robustness compared to running the defence against a well known suite of defence techniques beforehand. The adversaries will have gray-box knowledge of the ML systems. That is, attackers will know the training data the model was trained on and can use model gradients to create adversarial perturbations. However, they do not have any knowledge about the actual defence technique or the model architecture.

The SAFAIR AI contest investigates to motivate having more robust models in addition to designing more powerful and general applicable adversarial attacks.

3.2 Dataset

In the SAFAIR AI contest, attacks against and countermeasures in ML-based classifiers will be evaluated, and particularly the more specific task of Face Recognition is proposed. For this purpose, we use the CelebA dataset [15] to train the models. The CelebA dataset is a large-scale dataset of more than 200k celebrity images. Each of the images is annotated with 40 facial attributes. The images are focused on celebrity faces and consist of 10K unique identities with 40 binary attributes per image. CelebA is publicly available. We will release a development toolkit (when the contest starts) with a PyTorch[1] dataloader to simplify access to the data. We expect classification models to

[1] https://pytorch.org/

be trained on CelebA. The development toolkit also consists of PyTorch code for baseline models. For testing, the images will be chosen by the contest organisers. We have collected 1000 test images which are similar to the training dataset. The 1000 test images will be kept secret till the end of the contest. Participants should make use of only the CelebA dataset and the publicly available "train-val-test" split to train their models. The development toolkit enables easy access to the various splits and the training pipeline for the model.

3.3 Tasks and contest rules

We propose four (two attacks and two defences for each attack) different tracks for the contest:

1. **Targeted Face Re-Identification**. In this track, participants are given a set of face images and target identities. The purpose of the targeted face re-identification attack is to modify the input image in order to classify the image in a particular class label.
2. **Face Attributes Alteration**. In this track, participants are given a set of face images and a k-number of features ids. The purpose of the face attributes alteration attack is to modify the input image, but the k-features specified should be classified wrongly.
3. **Defence against Attribute Alteration.** In this track, participants design models robust to perturbations for face attribute alterations. The purpose of this task is to create a machine learning model which is robust to adversarial perturbations in the attribute alteration scenario. For instance, detecting adversarial images accurately.
4. **Defence against Targeted Face Re-Identification.** In this track, participants design models robust to perturbations for face re-identification. The purpose of this task is to create a machine learning model which is robust to adversarial perturbations to cause the model to classify the sample image as the particular target class.

3.4 Evaluation metrics

All evaluations are made based on the **L infinity norm**.

Attack: For the Attack team, the intention is to decrease the accuracy of the model. Let M be the model and S be the set of hidden Test samples. We run the model against these samples and compute its initial accuracy $A\_initial$. Then we run the same model M on the same dataset S against the participant's attack method T and compute the decrease in the accuracy. Let us say that the new accuracy is A_final. Then, the score for method T is $delta = A\_initial - A\_final$. The attacks are ranked based on the delta value highest first**.**

Defence: For the defence team, the intention is to suffer a minimum decrease in its initial accuracy as reported on unperturbed samples. Let M be the model and S be the set of hidden Test samples. We run the model against these samples and compute its initial accuracy $A\_initial$. Then we run the same model M on the same dataset S against the participant's attack method T and compute the decrease in the accuracy. Let us say that the new accuracy is A_final. Then, the score for method T is $delta = A\_initial - A\_final$. The defence teams are ranked based on the delta value smallest first.

The contest consists of two rounds. We have an initial round which enables us to select the top-5 best attack and defence methods.

3.4.1 Initial rounds

We follow the following steps to select the top-5 attacks and top-5 defences models:

- **Attack:** We have a baseline model trained based on Adversarial training using FGSM attack. This model would be pitted against the attack-submission $\boldsymbol{T}$. We will evaluate the decrease in the accuracy of the model based on the attack $\boldsymbol{T}$. For instance, if the original accuracy of the model on the task was 80% and the perturbed samples decreased it to 60%, the delta score is 20. Based on this delta value, the top 5 submissions would be selected. These would be our finalists for the attack configuration.
- **Defence:** For each defence method $\boldsymbol{D}$ applied to the initial model $\boldsymbol{M}$ we obtain $\boldsymbol{M'}$ (improved model) and the defence evaluation is made on how well $\boldsymbol{M'}$ behaves against FGSM, BIM, and PGD methods. We record the delta value and select top-5 teams based on minimum delta values. Since we test the model against three attack techniques in the initial round, we take a weighted sum of the delta value. Therefore, we assign FGSM, BIM, and PGD a weight coefficient of 0.2, 0.4, and 0.4, respectively. For instance, If the model has an initial accuracy of 100%, FGSM causes accuracy to go down to 80%, BIM causes to become 60%, and PGD causes to become 20%, then the final delta value is:

$$Final\,\Delta = (100 - 80) * 0.2 + (100 - 60) * 0.4 + (100 - 20) * 0.4 = 4 + 16 + 32 = 52$$

The second round (final evaluation) is a top-5 attacks vs top-5 defences evaluation.

3.4.2 Final evaluation

- **Attack:** The top-5 attacks are pitted against the top-5 defence methods. For each attack team T, we take the mean value of their delta score across the top-5 defence model. The winning team is the one having the **highest** mean delta value.
- **Defence:** Each defence method D (that creates M') is pitted against each of the top-5 attack methods. For each model M', we take the mean value of their delta score across the top-5 attack methods. The winning team is the one having the **lowest** mean delta value.

3.5 Contest schedule

The proposed schedule for the contest is the following:

- **February, 2021.** Launch the website with the announcement and contest rules. Start an active advertisement for the contest.
- **March 1, – May 15, 2021.** The contest will start in the first of March. Participants are working on their solutions. In the meantime, we organize few intermediate rounds of evaluation.
- **May 15, 2021.** Deadline for the final submission.
- **May 15 – May 31, 2021.** Organisers evaluate submissions.
- **May 31, 2021.** Announce contest results and release evaluation set of images.

3.6 Organizational aspects

The contest will be hosted on the Technical University of Munich website and/or GitLab[2]. Participants shall submit their solutions as Docker images (see 6). Submissions will be evaluated during the contest period, and the leaderboard will be updated consequently. When the contest finishes, we will conduct a final evaluation to decide each track winner.

3.6.1 Submission Process

Participants are expected to submit their submissions in docker images format. We will publish the specification in detail, how to create docker images and providing some simple examples in the contest development kit.

[2] https://about.gitlab.com

3.6.2 Rules

The main rules for the contest are the following:

1. TUM employees cannot participate in the contest.
2. You must register for the contest with one valid email address. If you participate in the contest with multiple email addresses, you will be disqualified.
3. The registration times are listed in the schedule section. If you register after the time periods, your evaluation will not be considered.
4. The contest is divided into four separate tasks, Targeted Face Re-Identification (Attacks and Defence) and Face Attribute Alterations(Attacks and Defence). Participants can be part of either the Attack track or Defence track but not both. However, participants for the Attack track of Face Attribute Alteration can compete for the Defence track of Targeted Face Re-Identification (but not for Defence track of Face Attribute Alteration), and vice versa.
5. Participants have to provide their submissions as open-source to the contest committee.
6. A model submitted by the defence team should be capable of handling valid inputs. In case the model fails to classify the input due to some error in the model, we penalize the model by considering it as a failure to handle an adversarial example.
7. The Attack team should produce an adversarial example in the gray-box setting. They can use model gradients to create adversarial perturbations. If the execution fails for a valid input example, due to some logical errors in the submission, we penalize the Attack team by skipping that example and assuming that the defence team was able to handle the perturbation without any misclassification.
8. Although we use a hidden test set for evaluation, still each classifier must be stateless. It should not memorize the training images.
9. The defence models should not produce random outputs for the same input at different points of time. Participants should make sure that their models are deterministic.
10. The number of submissions is limited to three times throughout the contest period (we will select the best submission from the three).
11. We have 1000 test samples which should be perturbed within 4 hours. If the time duration is exceeded, the perturbation process is interrupted and the team is penalized.
12. The Attack teams will be ranked based on the amount of decrease in accuracy they cause. This ranking will be based on the highest value first. defence teams will be ranked based on the decrease in their accuracy. The team which has maximum decrease will be placed last while the team having a minimum decrease in accuracy is ranked first. A separate ranking will be prepared for each of the four sub-tasks.

3.6.3 Documentation and tutorial

We will provide a comprehensive development kit as a package including various examples for submissions, tutorials, and evaluation:

- Examples for submission of models and attacks and a tutorial on how to work with them.
- The codes for baseline models and attacks with an extensive description.
- An example script to run attack on images and then run defence model on generated adversarial images, so participants will be able to get an understanding of whether their attack is good or bad by themselves.

Some of the above-mentioned bullet points will be reused from former AI contests and resources.

3.6.4 Contest dissemination

The contest will be promoted by TUM as the organizer, SAFAIR program members, SPARTA community and various university mailing lists.

Chapter 4 Adversarial machine learning robustness

Evaluation of machine learning models against adversarial examples is non-trivial. Research has shown that methods designed to create models which can withstand such adversarial attacks have a hard time. Only a few have shown some sort of robustness and most of the proposed defence techniques have shown to be incorrect [16]. We believe the most crucial element is the complexity of performing security evaluations. However, the research on adversarial attacks is grown quickly, the progress on defences is relatively slow. The reason for the slow advancement is due to the fact that most defences are instantly demonstrating incorrect or incomplete evaluation performances [17] [18] [19] [20] [21] [22] [23]. Therefore, researching this field and determining serious progress becomes very difficult.

As such, the SAFAIR program suggests, along with creating the contest, providing and utilizing any tools and methods that can aid in creating more robust machine learning models. The idea is to have a standardised benchmark to enable computing model performance in adversarial setting. We are especially interested in attack methods which can fool the system with minimum perturbations to the actual inputs. The SAFAIR benchmarking robustness solution will provides various reference implementations of adversarial attacks that can help the developers creating more defence techniques against the adversarial examples. In addition, it will demonstrate a benchmark report for the model robustness against suck attacks.

4.1 Motivation for the evaluation of the machine learning robustness

Since there are lots of solid arguments to investigate defences to the adversarial attacks, in the following, there are three basic reasons that developers and researchers getting interested in to evaluate the robustness of AI models [16].

- **Defending the systems against any adversarial attacks.** Adversarial examples are a crucial worrisome from a security viewpoint. Similar to any other new technology, developing ML systems are not considered by security from scratch. This may initiate the journey for adversaries to severely cause any harm (i.e., they benefit from the system misbehaving). With more and more machine learning models deployed in actual production code, the risks are high. In addition, since the models often times work on raw inputs obtained say, from sensors, it is difficult to predict the kind of harm an adversary can produce. For instance, an adversary tries to modify the decision made by a self-driving car to detect road signs incorrectly [24], cause a malware detection classifier to recognise a malicious binary as benign [25], cause an ad-blocker to classify an advertisement inaccurately as normal [26], or cause a recommender system to identify its suggestions incorrectly [27].
- **Testing the robustness of machine learning models in the worst-case scenario.** Since the machine learning models are deployed in real-life use-cases, they may come up against inputs that have an inherent randomness associated with them. In such scenarios, it is not easy to predict if the model would fail to behave in the expected manner. It is difficult to have an exhaustive coverage of these edge cases by means of testing. However, by analysing model performance in adversarial setting wherein an adversary is intent on causing a model failure, we can measure the worst-case robustness in real world setting.
- **Measuring the progress of ML methods in regards to the level of human capabilities.** Nowadays, we are achieving a very close performance of ML methods to humans. For instance, recent deep learning models are really good in tasks of natural language processing, reinforcement learning (e.g., Go and Chess [28]) or natural image classification [29]. However, when we take a look at adversarial examples which are not able to fool humans but can cause a huge performance dip for the machine learning models, this indicates a fundamental difference in the decision-making process of humans and machines.

The advances made in the fields of adversarial machine learning can aid us in advancing machine intelligence and can provide meaningful direction to the research field.

4.2 Theoretical verification of machine learning robustness

The verification of machine learning models is difficult and at the beginning of the pathway. Current approaches mostly focus on ensuring that the model assigns the same class label to all points within the specified neighbourhood of the training example x.

Scientists are researching intensively to develop verification systems for deep learning models [30]. Pulina et al. developed Pulina L. and Tacchella proposed the first verification system showing that the resulting classification is constant throughout the intended neighbourhood [31]. However, the initial verifications were performed on a relatively shallow network. It was verified for a neural network with a single hidden layer and less than twelve hidden units. To counter the problem of vanishing gradients, the sigmoid activation function is approximated using limitations.

However, modern deep learning models have millions of parameters that are substantially deeper and more complex. To provide verification for such systems, Huang et al. showed a better improvement regarding the first method and developed an updated verification system compatible with the modern neural networks [32]. However, to make the verification problem tractable, the method relies on a series of assumptions such as only a subset of hidden layer units are relevant for the input. Hence, with these limitations, the verification process cannot demonstrate an absolute guarantee not having any adversarial examples. Any adversarial example which violates these assumptions cannot be tracked by the verification process. For instance, a manipulation performed on one of the hidden units which were initially assumed to be irrelevant can produce an adversarial example capable of evading the verification system.

However, most of the modern deep learning-based models make use of rectified linear units (ReLU) [33] [34] [35]. The ReLU activation function is having a piecewise linear structure. Hence, by employing the ideas of constraint relaxation and using LP solvers, verification systems such as Reluplex [36] can provide some theoretical guarantees.

However, the verification systems mentioned above are restricted in scope since they just verify some particular neighbourhood of some definite point that the resulting output classification is still constant. The authors of challenge of verification [30] stated these two arguments:

1. “We do not have a way to exhaustively enumerate all x points near which the classifier should be approximately constant (we cannot imagine all future naturally occurring inputs).”
2. “The neighbourhoods surrounding x that we currently use are somewhat arbitrary and conservative; we tend to use L^p norm balls of small size because human observers agree that, for a small enough norm ball, all enclosed points should have the same class, but in practice, the region of the constant class should presumably be larger and have a different, less regular, less readily specified shape.”

In summary, ML models verification initially needs us to concretely determine the set of valid inputs, for instance, the inputs that we want our model to classify accurately. These valid inputs are usually significantly greater than the "testset" involved in most benchmarks. Scientists should develop verification methods that be able to accurately ensure the ML predictions created throughout the whole valid inputs. The main goal of developing machine learning models is to build models that can generalize to new data inputs. However, oftentimes the generalization requirement may not align perfectly with the verification task [31] [32] [36], making the task more challenging. In this case, other researchers' methods might allow incomplete verification of ML models during procedures closer to the testing—for example, a positive influence of fuzzing in the cybersecurity domain.

We should always be aware that no verification system ever exists that guarantees the full robustness of machine learning models. Especially the challenge of generalising the ML models to new input data that was not seen before.

In traditional ML systems, we have definite fundamental restrictions on the results of ML classifiers for new unseen data points. For instance, the well-known theorem “no free lunch” (NFL) [37] implies that there is no single ML algorithm that universally performs the best for all problems.

We can apply the NFL theorem in adversarial settings too. The contest between attackers and defenders may conduct into two feasible outcomes. One could be kept in mind that the attackers always one step ahead of the defenders due to inherent statistical complexities connected to predicting new unseen test data, or the defenders have primary privileges due to large amount of problems. If we are fortunate fairly, the second scenario may come true and could help a lot in developing and verifying ML methods to ensure robustness.

Nicolas et al. [38] describes the trade-off between model accuracy and robustness to adversarial attacks. It reveals that if an attacker is able to find an underlying distribution that rises the loss in the learner, afterwards, the learner has the advantage of changing into a stronger and better hypothesis class. The better hypothesis class can be the more complex class that demonstrates the minimum loss at any underlying distribution. Subsequently, a potential tension appears in lacking enough data because a more complex hypothesis learner needs more data.

4.3 Best practices to evaluate adversarial robustness

To help developers and scientists having guidelines in performing the evaluations of adversarial robustness, we have got inspired by developed checklists, best practices, and methods that list current evaluation pitfalls in this domain [16]. The sections mentioned below are designed by to list common evaluation flaws. We are also aware that these lists are not complete enough to perform an evaluation on model robustness but they could help for recognising possible evaluation flaws. The checklists are compiled and developed by the authors [16] in the GitHub repository[3] too to check for details.

4.3.1 Common severe flaws

Various common critical evaluation flaws may lead to revoking any robustness. We have got the sections 4.3.1, 4.3.2, and 4.3.3 from the authors of evaluating adversarial robustness [16] that had extensive practical efforts on adversarial model robustness:

- “State a precise threat model that the defence is supposed to be effective under:
    - The threat model assumes the attacker knows how the defence works.
    - The threat model states attacker’s goals, knowledge and capabilities.
    - For security-justified defences, the threat model realistically models some adversary.
    - For worst-case randomized defences, the threat model captures the perturbation space.
- Perform adaptive attacks to give an upper bound of robustness.
    - The attacks are given access to the full defence, end-to-end.
    - The loss function is changed as appropriate to cause misclassification.
    - Focus on the strongest attacks for the threat model and defence considered.
- Release pre-trained models and source code.
    - Include a clear installation guide, including all dependencies.

[3] https://github.com/evaluating-adversarial-robustness/adv-eval-paper

- There is a one-line script which will classify an input example with the defence.
- Report clean model accuracy when not under attack.
  - For defences that abstain or reject inputs, generate a ROC curve.
- Perform basic sanity tests on attack success rates.
  - Verify iterative attacks perform better than single-step attacks.
  - Verify increasing the perturbation budget strictly increases attack success rate.
  - With high distortion, model accuracy should reach levels of random guessing.
- Generate an attack success rate vs. perturbation budget curve.
  - Verify the x-axis extends so that attacks eventually reach 100% success.
  - For unbounded attacks, report distortion and not success rate.
- Verify adaptive attacks perform better than any other.
  - Compare success rate on a per-example basis, rather than averaged across the dataset.
  - Evaluate against some combination of black-box, transfer, and random-noise attacks.
- Describe the attacks applied, including all hyperparameters."

4.3.2 Common pitfalls

- "Apply a diverse set of attacks (especially when training on one attack approach).
- Try at least one gradient-free attack and one hard-label attack.
- Perform a transferability attack using a similar substitute model.
- For randomized defences, properly ensemble over randomness.
- For non-differentiable components, apply differentiable techniques.
- Verify that the attacks have converged under the selected hyperparameters.
- Carefully investigate attack hyperparameters and report those selected.
- Compare against prior work and explain important differences."

4.3.3 Special-case pitfalls

- "Investigate if it is possible to use provable approaches.
- Attack with random noise of the correct norm.
- Use both targeted and untargeted attacks during evaluation.
- Perform ablation studies with combinations of defence components removed.
- Validate any new attacks by attacking other defences.
- Investigate applying the defence to domains other than images."

4.4 A tool to benchmark adversarial machine learning solutions

We have already seen that verification for machine learning models is difficult. This is even worse for deep learning models which tend to more prevalent in the current use-cases. However, even straightforward testing is tricky. For instance, a researcher proposes a new defence mechanism. He tests it against a particular adversarial attack and obtains a high accuracy value. Now, does this imply that the new defence is robust? Maybe but there is also a non-zero probability of the attack implementation being weak to begin with. A similar problem happens when a researcher proposing a new attack is testing his implementation against the well-known defence techniques.

To resolve these difficulties, we plan to develop a method and tool to benchmark adversarial machine learning solutions with their results which will be the next deliverable D7.6. This tool contains reference implementations of several attacks and defence procedures. We will also reuse different open-source solutions such as Cleverhans [39] and Foolbox [40] for our purpose.

The implementation of attack mechanisms can be used for two main tasks. Firstly, it can be employed during adversarial training. Adversarial training is currently the most effective method in adversarial setting. It involves generating adversarial perturbations during the training procedure and hence, the reference implementations can be useful. Secondly, by means of providing reference implementation of various attack mechanisms, we have a tool which can aid in benchmarking. Due to the need for a standard reference implementation, we cannot compare different benchmarks. A benchmark resulting in high accuracy may indicate more robustness; however, it may additionally show that the attack implementation is weak too. By using the proposed tool, our researchers, especially in the SAFAIR program, can be ensured that the reporting high accuracy on our benchmarking approach corresponds to a robust model.

Besides, developers and researchers are able to utilize our proposed tool to evaluate the robustness of their proposed solutions against standardised, state-of-the-art attacks and defences. Then, if a defence demonstrates a top score accuracy against the tool attack, the evaluation conclusively indicates that the defence defeats this standardised implementation of attacks; on the other hand, if an attack demonstrates a top score failure rate against a tool defence, the evaluation conclusively indicates that the attack is being capable of defeating a definite implementation of the defence.

Core functionalities

The adversarial ML benchmark tool will be a modular architecture oriented to code reuse. It will consist of these primary modules:

- **Attacks:** this module contains various sample attack implementations. One can use the attack class module as a template and easily implement for other open-source tools such as Foolbox [40] adversarial attacks.
- **Model:** contains the source code for a PyTorch model. Please note that we expect that the model is already trained and we are going to test it against adversarial perturbations.
- **Wrapper:** contains the model converter. This would take a model, for example, implemented in PyTorch and convert it into other open-source tools, such as Foolbox [4].
- **Dataset:** contains the code for loading different dataset.
- **Use cases:** Contains the logic specific to different use cases such as the Face Re-identification and Face Attribute Alteration tasks in the SAFAIR AI Contest.

[4] https://github.com/bethgelab/foolbox

Chapter 5 Summary and Conclusion

This document proposed two main solutions to evaluate the adversarial machine learning methods within the SPARTA WP7 SAFAIR program. One is designing an adversarial machine learning contest to test participants' attacks and defences solutions against each other which is an intermediate solution. The second one is implementing an adversarial ML benchmark tool that helps researchers and developers to design and implement more robust ML models and present standardised benchmarks of proposed solutions in the area of adversarial machine learning. Besides, we have provided some methods and suggestions for evaluation of the model robustness. However, these two proposed solutions are generic and adaptive, which can be used for lots of different adversarial ML scenarios (not just SAFAIR ML scenarios) for evaluations.

To this end, the verification of ML models robustness is at the beginning of the pathway because the algorithms and techniques have presumptions that avoid them presenting full guarantees of not having any adversarial examples. Consequently, we hope our readers will be inspired to solve some of the challenges mentioned in this document.

Chapter 6 Bibliography

[1] I. Corona, B. Biggio, and D. Maiorca, “AdversariaLib: An open-source library for the security evaluation of machine learning algorithms under attack,” *arXiv Prepr. arXiv1611.04786*, 2016.

[2] A. Kurakin *et al.*, “Adversarial attacks and defences competition,” in *The NIPS’17 Competition: Building Intelligent Systems*, Springer, 2018, pp. 195–231.

[3] R. M. Williams and R. V Yampolskiy, “Optical Illusions Images Dataset,” *CoRR*, vol. abs/1810.0, 2018, [Online]. Available: http://arxiv.org/abs/1810.00415.

[4] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” 2015.

[5] A. Kurakin, I. J. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” 2019.

[6] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” 2018.

[7] N. Carlini and D. Wagner, “Towards Evaluating the Robustness of Neural Networks,” 2017, doi: 10.1109/SP.2017.49.

[8] F. Tramèr and D. Boneh, “Adversarial training and robustness for multiple perturbations,” in *Advances in Neural Information Processing Systems*, 2019, pp. 5866–5876.

[9] A. N. Bhagoji, D. Cullina, C. Sitawarin, and P. Mittal, “Enhancing robustness of machine learning systems via data transformations,” 2018, doi: 10.1109/CISS.2018.8362326.

[10] X. Echeberria-Barrio, A. Gil-Lerchundi, I. Goicoechea-Telleria, and R. Orduna-Urrutia, “Deep Learning Defenses Against Adversarial Examples for Dynamic Risk Assessment,” in *Conference on Complex, Intelligent, and Software Intensive Systems*, 2020, pp. 316–326.

[11] Z. Katzir and Y. Elovici, “Detecting Adversarial Perturbations through Spatial Behavior in Activation Spaces,” 2019, doi: 10.1109/IJCNN.2019.8852285.

[12] M. Pawlicki, M. Choraś, and R. Kozik, “Defending network intrusion detection systems against adversarial evasion attacks,” *Futur. Gener. Comput. Syst.*, 2020, doi: 10.1016/j.future.2020.04.013.

[13] T. Weber, S. Conchon, D. Déharbe, M. Heizmann, A. Niemetz, and G. Reger, “The smt competition 2015--2018,” *J. Satisf. Boolean Model. Comput.*, vol. 11, no. 1, pp. 221–259, 2019.

[14] W. Brendel *et al.*, “Adversarial vision challenge,” *arXiv Prepr. arXiv1808.01976*, 2018.

[15] N. Ateqah, B. Mat, N. Hidayah, B. Abd, and Z. Ibrahim, “Celebrity Face Recognition using Deep Learning,” vol. 12, no. 2, pp. 476–481, 2018, doi: 10.11591/ijeecs.v12.i2.pp476-481.

[16] N. Carlini *et al.*, “On Evaluating Adversarial Robustness,” *arXiv Prepr. arXiv1902.06705*, 2019.

[17] N. Carlini and D. Wagner, “Defensive distillation is not robust to adversarial examples,” *arXiv Prepr. arXiv1607.04311*, 2016.

[18] A. Nayebi and S. Ganguli, “Biologically inspired protection of deep networks from adversarial attacks,” *arXiv Prepr. arXiv1703.09202*, 2017.

[19] N. Carlini and D. Wagner, “Adversarial examples are not easily detected: Bypassing ten detection methods,” 2017, doi: 10.1145/3128572.3140444.

[20] W. He, J. Wei, X. Chen, N. Carlini, and D. Song, “Adversarial example defenses: Ensembles of weak defenses are not strong,” 2017.

[21] N. Carlini and D. Wagner, “Magnet and" efficient defenses against adversarial attacks" are not robust to adversarial examples,” *arXiv Prepr. arXiv1711.08478*, 2017.

[22] C. Cornelius, N. Das, S.-T. Chen, L. Chen, M. E. Kounavis, and D. H. Chau, “The Efficacy of SHIELD under Different Threat Models,” *arXiv Prepr. arXiv1902.00541*, 2019.

[23] N. Carlini, “Is AmI (attacks meet interpretability) robust to adversarial examples?,” *arXiv Prepr. arXiv1902.02322*, 2019.

[24] N. Papernot, P. Mcdaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, “The limitations of deep learning in adversarial settings,” 2016, doi: 10.1109/EuroSP.2016.36.

[25] G. Dahl and J. Stokes, “Large-scale malware classification using random projections and neural networks,” *Acoust. Speech …*, 2013, doi: 10.1109/ICASSP.2013.6638293.

[26] F. Tramèr, P. Dupré, G. Rusak, G. Pellegrino, and D. Boneh, “Ad-versarial: Defeating Perceptual Ad-Blocking,” *CoRR*, vol. abs/1811.0, 2018, [Online]. Available: http://arxiv.org/abs/1811.03194.

[27] T. Di Noia, D. Malitesta, and F. A. Merra, “TAaMR: Targeted Adversarial Attack against Multimedia Recommender Systems,” 2020, doi: 10.1109/DSN-W50199.2020.00011.

[28] D. Silver *et al.*, “Mastering the game of Go with deep neural networks and tree search,” *Nature*, 2016, doi: 10.1038/nature16961.

[29] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” 2012.

[30] Ian Goodfellow and Nicolas Papernot, “The challenge of verification and testing of machine learning.” http://www.cleverhans.io/security/privacy/ml/2017/06/14/verification.html.

[31] L. Pulina and A. Tacchella, “An abstraction-refinement approach to verification of artificial neural networks,” in *International Conference on Computer Aided Verification*, 2010, pp. 243–257.

[32] X. Huang, M. Kwiatkowska, S. Wang, and M. Wu, “Safety verification of deep neural networks,” 2017, doi: 10.1007/978-3-319-63387-9_1.

[33] V. Nair and G. E. Hinton, “Rectified linear units improve Restricted Boltzmann machines,” 2010.

[34] X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier neural networks,” 2011.

[35] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, “What is the best multi-stage architecture for object recognition?,” 2009, doi: 10.1109/ICCV.2009.5459469.

[36] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, “Reluplex: An efficient smt solver for verifying deep neural networks,” 2017, doi: 10.1007/978-3-319-63387-9_5.

[37] D. H. Wolpert, “The Lack of a Priori Distinctions between Learning Algorithms,” *Neural Comput.*, pp. 1341–1390, 1996, doi: 10.1162/neco.1996.8.7.1341.

[38] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman, “Towards the Science of Security and Privacy in Machine Learning,” Nov. 2016, Accessed: Sep. 24, 2020. [Online]. Available: http://arxiv.org/abs/1611.03814.

[39] N. Papernot *et al.*, “CleverHans v2.1.0: an adversarial machine learning library,” *arXiv*, 2018.

[40] J. Rauber, W. Brendel, and M. Bethge, “Foolbox: A Python toolbox to benchmark the robustness of machine learning models,” *arXiv*. 2017.