POLITECNICO DI TORINO

III Facoltà di Ingegneria

Corso di Laurea Magistrale in Computer Engineering

Master Thesis

Evasion attacks against machine learning-based behavioral authentication

Advisors:
Prof. Paolo Garza
Prof. dr. ir. W. Joosen
Dr. ir. D. Preuveneers

Candidate:
Marco Farinetti

Academic Year 2017-2018

Summary

Authenticating a user means verifying his identity. Today, authentication systems are used for all kind of applications, in various forms: from traditional secrecy-based methods to more modern biometrics-based ones. Behavioral authentication in particular, has become very relevant as a mean of continuously verifying a user's genuineness. These systems are built on top of machine learning algorithms, and as such, they are subject to adversary attacks. Evasion attacks rely on generating adversarial instances capable of evading a classifier with small modifications. While adversarial instances have been successfully generated for differentiable models, this is not true for tree ensembles, for which the literature is very limited.

In this work we evaluate the resilience of a gait-based authentication system, based on tree ensembles, against evasion attacks. First, we propose an algorithm for generating adversarial instances while constraining them to be hard to detect. Secondly we define metrics for evaluating the evasion performance. Finally, we deploy a defensive mechanism to detect adversarial instances before they are submitted to the tree ensemble. We validate our research by testing the attack in multiple scenarios, using a dataset of gait data collected from 17 subjects and 9 body positions.

We show that adversarial instances can be generated starting from the adversary own data. Our analysis shows that instances generated in this way are hard to detect, with evasion rates between 60% and 90%, depending on the constraints imposed during the generation. Our defensive mechanism is able to reject up to 40% of the adversarial instances generated.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Authentication is a concept relevant to multiple fields. In computer science, it means verifying a person's identity to allow access to confidential information. Throughout history, different types of authentication systems have been deployed, but traditionally the most widely used are secret-knowledge based. Passwords are an example of secret knowledge. Typically, secret-knowledge based authentication systems authenticate users only at the entry point of application, which can be cause of vulnerability. Continuous authentication systems address this issue by constantly re-authenticating the user. Such systems must be transparent, which means they should not interfere with the user's activity; behavioral authentication is particularly suited for this scope, and has become very relevant for providing a mean of continuously verifying a user's genuineness.

Just like secret-knowledge based systems are vulnerable to brute-force or guessing attacks, also continuous systems can be attacked. Since these systems are often based on biometrics, defensive measures like liveliness checks and anti-spoofing mechanism proved to be effective against traditional attacks. However, these systems often exploit machine learning techniques to learn the biometric and behavioral traits of users and to authenticate them. This leads to the presence of another type of attack, against which the listed measures are helpless: attacks targeting directly the learning algorithm. Machine learning models can be bypassed using poisoning or evasion attacks. We are interested in the second type, which exploits a weakness of these models: for virtually any instance x correctly classified by the model, it is possible to find a small perturbation δ such that $x + \delta$ is misclassified. These vulnerabilities, and the possible countermeasures, are the study subject of the adversarial machine learning field.

The objective of the thesis is to evaluate the resilience of machine-learning based authentication systems against evasion attacks. We conduct our study on a behavioral gait-based authentication system that relies on tree ensembles to recognize the users. While recent works has shown how evading instances can be generated

for differentiable models [5] [11], such as support vector machines and neural networks, very few studies [15] cover this topic for ensemble models, making them an interesting study subject.

In this work we review the state-of-the-art of gait-based authentication and recognition, and of adversarial machine learning, focusing on evasion attacks. Then, we define the system on which we conduct the experiments. The system is an adaptation of an existing system proposed by Van hamme et al. [22]. Finally, we model an adversary and define the possible attacks scenarios. The contributions of our study are:

- We define an algorithm for generating adversarial instances for tree ensembles, and constraining them to be hard to detect. We test the effectiveness of the generated samples in the context of behavioral authentication.
- We deploy a defensive mechanism to detect such adversarial instances. The system, based on outliers detection, performs an input sanity check with the goal of rejecting adversarial instances before they are submitted to the authentication system.
- We perform a security evaluation of a gait authentication system where an adversary performs evasion attacks by manipulating input data, in order to show the threat posed by this kind of attacks.

We evaluate the effectiveness of the proposed algorithm and defensive measures using data from the REALDISP open benchmark dataset [3], which contains the gait data of 17 subjects, collected from 9 different body positions. Finally, we discuss the results, the limitations of our approach, and how it can be improved in future work.

Thesis Structure

The thesis is structured in the following way:

- **Background** 2: Context and concepts on which the work is based are introduced. A reader familiar with these topics may skip the chapter.
- **Related Work** 3: Review of the state-of-the-art of gait-based authentication systems and adversarial machine learning.
- **System Design** 4: Description of the authentication process and design of the authentication system and of the defensive measures.

- **Attack Methodology** 5: Definition of the attack scenarios, modeling of the adversary and definition of the algorithm to generate adversarial instances.
- **Evaluation** 6: Testing and validation of the system. Experiments methodology and results.
- **Discussion** 7: Discussion of the results, of the contribution and of the limitations. Future work is suggested.

Chapter 2

Background

In this section we introduce the concepts on which our work is based, with the goal of providing the required knowledge and context to read this thesis. First, we provide an overview on the current state of authentication systems and their characteristics. In the second part we present adversarial machine learning, the branch of machine learning that evaluates security of machine learning techniques in adversarial environments.

2.1 Authentication Systems

Protection of IT systems is usually provided via user identification or Authentication, which subsequently enables successful Authorization and Accountability [1]. We refer to these concepts as AAA. Authenticating a user means verifying his identity, making sure that he is the legitimate user he claims to be. After a successful authentication, authorization is given according to a set of predefined permissions. Finally, accountability means keeping track of activities performed by the user. The focus of the thesis is the authentication process. Conventional approaches to authentication can be split in three categories:

- **Secret-knowledge based:** The most common are PINs and passwords. While the first is relatively easy to guess due to being short, the second one can be strong if chosen appropriately. Passwords, however, are vulnerable to misuse, which include using dictionary words, using the same password on different systems or writing them somewhere in plain text. Brute force attacks can be stopped by locking an account after a certain amount of failed login attempts.

- **Token-based:** Token-based authentication methods include both hardware solutions, such as bank smartcards, and software solutions, such as Google

authenticator. The reliance on the presence of the token provides compromise detection, since being used daily (e.g. smartphone) makes it easier to detect a loss. Tokens are generally used with other authentication factors, but present the disadvantages of being expensive (hardware solutions) and of requiring synchronization.

- **Biometrics:** Biometrics present lots of advantages compared to other solutions. They enable both authentication and verification, there is no memory involved in the process, and their non-transferrability characteristic makes them strong against repudiation. Biometric authentication does not output a boolean result, but rather a confidence measure based on error rate. Biometrics can be faked but liveness detection mechanisms can be put in place as defensive countermeasures.

All the listed techniques share a major weakness: they only protect the entry point of the application.

Continuous and transparent authentication systems have been developed with the goal of addressing the entry-point issue. Such systems are continuous because they periodically, or constantly, verify whether the user is legitimate using biometrics. They are transparent in the sense the the normal user experience isn't interrupted by the authentication process.

Transparent Authentication Systems (TAS) can be split in two categories: physiological systems, such as fingerprint and facial recognition-based ones, and behavioral systems. Implementing fingerprint and facial recognition in a transparent way proved to be difficult, therefore the focus shifted to behavioral counterparts. Behavioral techniques are, for example, keystroke and mouse dynamics, gait recognition and voice recognition. They are easier to implement in a transparent way but they still present some limitation. Such limitations are usually related to the systems they work on (some work only on mobile, others only on PC), and to their effective functionality in a universal and not confined system.

2.2 Adversarial Machine Learning

Machine Learning in an adversarial environment requires trying to anticipate the opponent moves [13]. An adversary tries to break the assumptions that practitioners make by either adapting the data to be similar to normal data or by poisoning the data i.e. crafting input data that when retrained on generates incorrect decision making functions.

Attacks can be classified according to what they influence and what kind of security property they try to violate. Causative attacks alter the training process of the system while exploratory attacks probe the detector to discover how it works

or to gather information about the training data. For the security violation aspect we have three categories:

- **Integrity:** Integrity violation results in malicious samples being classified as normal.
- **Availability:** Availability violation causes the system to produce many classification errors, both false positives and false negatives, and therefore becoming unusable.
- **Privacy:** Privacy violation results in the adversary obtaining information about the learner and as a consequence compromise the privacy of the system's users.

Attacks can be modeled as an attacker (A)/defender (D) game where D chooses a learning algorithm H and A can either attack the learning phase (A_{train}) or the evaluation phase (A_{eval}). If it chooses to attacks the learning phase, the training dataset is contaminated by A_{train}, and therefore the resulting decision function will behave in an unexpected way for the defender. On the other hand, evaluation phase attacks contaminate the evaluation dataset, leading to incongruences when comparing predictions to the true labels of the samples.

Modeling the adversary knowledge and capability is fundamental to evaluate an attack and to understand what kind of defense can be put in place.

Application-specific factors The application domain itself may pose some limits to an attacker. A defender should define how an adversary can interact with the application and what data he may be able to obtain or manipulate. Depending on these limitations, only some attacks may be feasible. An example of application-specific factor is the feature space used by the learning algorithm. Whether the features are binary or real-valued, the size of the feature space and how such features are computed are all factors that must be taken into account. While a binary value can only be flipped, a real-valued feature may be drastically changed. Finally, the distribution of the data may affect the performance of a model and therefore the ability of an adversary to attack it. For example, when dealing with sparse datasets, rare features may have more influence on the final decision, and therefore an attack targeting them may be more successful.

Modeling attacker capabilities There are many possible ways an attacker can influence the model, and they must be clearly defined. Can he alter existing samples or can he only insert new ones in the datasets? What samples does he have access to? How much can those samples be altered? Class limitations also need to be taken into account, for example an adversary may only be able to alter the positive or the negative class.

Modeling attacker knowledge Finally, it is necessary to define what the adversary knows about the system. The three key components of a learner are the learning algorithm, the data and the feature space, and attackers may have different degrees of information about them. The security of a system depends on whether these assumptions reflect the effective knowledge of the attacker. Unrealistic assumptions may make a system insecure, if its secrecy is overestimated, or impossible to protect in presence of an "omnipotent" adversary.

Generally, the learning algorithm is assumed to be known by the attacker. While the feature set may be kept secret, it is reasonable to assume that at least a part of it can be inferred by the adversary. This is due to some features being re-used in different learning tasks and to the fact that specially selected features may be approximated by simpler ones. As for the data, it is reasonable to make stronger secrecy assumptions. It is generally difficult for an adversary to have access to the training data, while it may be easier to insert samples in the evaluation dataset. Systems that use algorithms based on re-training should pay particular attention to this aspect.

Attack goals: Cost-based evasion While it may be relatively easy to alter an instance to make it evade detection, such instance may be less useful to the adversary. This is due to the "cost" of adversary's evasive actions. The attacker looks for the lowest cost instance (from his perspective) that evades the classifier. The adversary tries to optimize instances to have high-value while still evading detection, while on the system side, precautions should be taken to find those optimized instances. In the real world, it is difficult to find near-optimal evasion because, even if feature mapping is known, designing a single object that maps to a specific query is hard. However, it is also true that real world adversaries do not necessarily need near-optimal instances but finding an acceptably low-cost instance is often enough.

Chapter 3

Related Work

In this section we introduce studies related to the two core concepts of the thesis: gait authentication and adversarial machine learning.

First, we analyze the performance and the peculiar characteristics of authentication systems based on gait. In the second part of the chapter, we analyze the effectiveness of different attack techniques on various machine learning models. We cover both evasion and poisoning attacks on Support Vector Machines, Neural Networks and Tree Ensembles.

3.1 Gait-based authentication systems

Gait analysis, the study of human motion, is used in different fields: human identification [2], activity recognition [4] and authentication [14] [9] [8] [17] [22] [10]. Studies [7] have proved that it is possible to recognize an individual from their walking style, which indicates that gait carries identity information. It also appears that, when all components are considered, human gait is unique [18] and difficult to disguise. Given that an individual is uniquely identified by his gait, such trait can be used for authentication purposes. Gait data can be obtained from external sources, such as cameras [14] or floor sensors [16], or from sensors directly placed on the individual, specifically accelerometers [9] [8] [17].

To provide an example of how camera-based systems work, Jia et al. [14] use silouette contour analysis to perform the authentication task, computing the gait flow image and head and shoulder mean shape to represent a subject's gait. The similarity score chosen is the euclidean distance. Such a system can deal with view variation and produces very good results with 1.07% false acceptance rate (FAR).

In the context of accelerometer-based systems, the interesting topics are the quality of accelerometers used, the extraction of features, and the methods used to build the models. In the work of Gafurov et al. [10], data is gathered by an

accelerometer placed in the trousers pocket of the subjects. The model is based on the extraction of gait cycles, each made of 100 accelerometer values, leading to a 100 component feature vector. Similarity scores are computed according to absolute distance and correlation, producing an EER of 7.3%. Findling et al. [9] try to build simple machine learning models to be used on smart cards. In this case the acceleration data is obtained from mobile phones, therefore providing less accurate values. Again, data is processed to extract gait cycles and the results obtained are reasonable, with a 12% equal error rate (EER). Derawi et al. [8] use hidden markov models to authenticate the users. With the same dataset, the results obtained using a traditional distance metric and using the hidden markov models vary greatly: a 20% EER is observed in the first case, while a 10% EER is obtained in the second case. The latter also provide an important advantage: they remove the necessity of extracting features and work directly on the accelerometer data. This leads to better performance since it eliminates the errors due to gait cycle extraction. Thang et al. [17] extract time and frequency domain features from mobile phones built-in accelerometer data. The frequency domain features are used to train a support vector machine that performs a classification task. This approach achieves a 92.7% accuracy, which is very good when compared to other systems using mobile phone accelerometers.

The model used by Van hamme et al. [22] is built on a common set of extracted features, which is independent of the type of activity. The dataset used includes data from the different activities, obtained by both ideally-placed and self placed sensors. Both time and frequency domain features are extracted, resulting in a feature vector of length 224. A model is trained for each of the study subjects and each of the activities. The models are trained and tested on a dataset made partially of data belonging to the subject and partially of data belonging to other subjects, in equal proportion. The classification algorithm chosen is Gradient-Boosted Trees.

In a non-adversarial environment, the results obtained are very good, with equal error rates ranging from 2% to 8%, depending on the activity and body position considered.

3.1.1 Resilience against non-zero effort attacks

Van hamme et al. [22] try to address common issues concerning gait-authentication systems: resilience against sensor displacement, feasibility in uncontrolled environments and vulnerability to external attacks. The last point is the most interesting for our work.

The authentication system is tested in a scenario where an adversary tries to cause misclassification executing an observation attack. Obtaining the trace of a single accelerometer is feasible, e.g. when a smartphone sensor is used since accelerometer data can be collected through the HTML5 API. Therefore, a single

sensor system is not reliable from a security point of view. The proposed solution combines two sensors placed in different body positions to improve the resilience against this kind of attacks. The new feature vector has length 448.

In a zero-effort attack, the adversary uses his own data to authenticate as another user. This is not different from the basic testing scenario and, as previously stated, the results are very good. In the non-zero effort scenario considered, the adversary uses one trace acquired by observing the target user (e.g. the smartphone one) and one trace of his own data. It is also assumed that the attacker knows the position of the user's sensor. In this case, the results are significantly worse, with an EER of 17%. However, taking into account the eventuality of such an attack, the system can be trained to be more resilient by adding some attack samples to the training data. With said adjustments, the new EER is 4%, only slightly worse than before (3.7%).

The authentication system performs well when not all the sensors are compromised. However, the study does not cover more sophisticated attacks where the adversary may try to forge ad-hoc inputs, either by reproducing the user traces or by modifying his own data.

3.2 Attacks against Machine Learning

Adversarial machine learning is the study of machine learning techniques against adversarial opponents [13]. Adversarial machine learning requires studying what kind of attacks can be carried out by an adversary and what are his capabilities and limitations.

Huang et al. [13] define a taxonomy for attacks against machine learning systems, according to some properties. The most important property concerns how the attack influences the model: causative (poisoning) attacks target the training data to influence the model when re-training happens; exploratory attacks, instead, use different techniques to gain insight on the model and find vulnerabilities that can be exploited in the classification phase. Evasion attacks are part of this second subset.

3.2.1 Poisoning attacks

The goal of poisoning attacks is to influence the training data. An attacker may have different ways of influencing such data: for instance he can control a fraction of the training instances by submitting samples to the model. While it may be hard to influence the initial training data, many models do a re-training process on new instances, which include the ones submitted by the adversary. The learner will therefore produce a bad classifier, which the adversary can exploit.

An example of such attacks is the *correlated outlier attack* tested by Newsome et al. [19] against a virus detector. The adversary generates positive training instances and adds some spurious features, which cause benign traffic with those features to be blocked.

Poisoning attacks have also been successfully carried out against face recognition systems [6]. Such systems update their models to account for natural changes over time and therefore can be vulnerable to such attacks. B. Biggio [6] shows that by presenting a carefully crafted sequence of inputs to the system, the face templates can be compromised. Eventually, the adversary can even impersonate a user or cause the system to deny access to them.

3.2.2 Evasion attacks

Evasion attacks are the most prevalent type of attacks encountered in adversarial settings. The goal of the adversary is to evade the classifier without direct influence over the classifier itself. There are different kind of evasion attacks: some try to emulate the properties of normal samples in order to hide intrusions; other attacks, instead, rely on crafting adversarial examples, instances that exploit weaknesses of the model causing it to misclassify. There has been lots of success in generating adversarial instances for differentiable models [11], but the same is not true for tree ensembles.

Szegedy et al. [21] show that several machine learning models, state-of-the-art neural networks included, are vulnerable to adversarial examples. Such examples differ very slightly from correctly classified instances, such as those belonging to the training set, but they are misclassified by the model. In many cases, different models trained on the same dataset can be evaded with the same adversarial examples (transferability property). Crafting adversarial examples require finding the worst-case perturbations that can be applied to examples from the original dataset. Such perturbed examples are then misclassified with high confidence.

SVMs and Neural Networks

Biggio et al. [5] test a gradient-based approach against neural networks and SVMs. The testing scenario is again malware detection in PDF files. The adversary is characterized by what he knows, e.g. training set, feature representation, learning algorithm and model or feedback from the classifiers, and what he can do, e.g. modify the input data, the feature vectors, or specific features independently. An attacker with "perfect knowledge" has access to all of the above and he is only constrained by a maximum distance from the original sample. An attacker with "limited knowledge" instead, only knows the feature representation and the type of the classifier and he is still limited by a maximum distance parameter. He can learn

a new model from a surrogate dataset, which gives him an approximation of the discriminating function. Obtaining the surrogate dataset is application dependent but it is assumed to be a feasible task. In the malware detection example, the surrogate model is built using 500 malicious samples from the Contagio dataset and 500 benign samples from the web. The experiment is performed in both the perfect knowledge (PK) and limited knowledge (LK) scenarios. The performance of the two is very similar, with the LK case requiring slightly higher variations from the original sample. The basic gradient-descent attack evades the SVMs with few modifications, but it gets stuck at ~20% false negative rate against the neural network. Introducing a "mimicry" parameter, which makes the descent sub-optimal but helps to avoid local minima, makes the attack work also against the neural network. However, in this scenario, the difference between the LK and the PK case is evident, with the LK adversary achieving only a ~60% FN rate.

Xu et al.[24] test evasion attacks against PDF malware classifiers. Starting from a malicious sample, the goal is to manipulate it such that *(i)* it preserves its maliciousness and *(ii)* it's classified as benign. The two PDF malware classifiers chosen are PDFRate, which uses random forests, and Hidost, which uses SVMs. The adversary is modeled as follows: he has no detailed knowledge of the classifier and of the features used, however he can access a black-box of the model that returns the classification score, he is in possession of some malicious samples and has general knowledge of benign samples. The proposed approach is based on genetic programming, generating variants of the malicious sample until an evasive one is found. 500 malicious samples are taken from the Contagio PDF malware archive for testing the attack. Elusive variants are successfully found for all of them, taking on average 16 minutes per sample on PDFRate and 5 minutes per sample on Hidost. Obtaining such adversarial instances requires many iterations, therefore this approach is better suited to attack offline models.

Tree Ensembles

There are not many studies about tree ensembles in adversarial settings, especially in regard to generating adversarial examples. Kantchelian et al. [15] propose two approaches for generating evading instances for such models, an optimal and an approximate one. The attacks are tested on a binary classification task. The goal is to distinguish between handwritten digits "2" and "6". The learner used is a gradient boosted ensemble with one thousand trees of maximal depth 4, and the number of dimensions n is 784. The effectiveness of the attack is quantified as "attack effort" i.e. the amount of perturbation needed to obtain an evading instance. The results vary depending on the distance metric, with the L_∞-distance being the best performing one, requiring a median perturbation of 0.06%.

Chapter 4

System Design

In this chapter we describe the design and the working principle of the system we are studying. We start by showing the process of authenticating a user at a high level. Secondly, we describe the input sanity check and how we built it. Finally, we define the classifier on which the gait authenticator is based and what transformations are applied on it to suit our study case.

4.1 Authentication Process

Here we present the authentication system which is the subject of our study. Users of the system are authenticated based on their gait data, which is collected by a sensor placed on them. The sensor can be placed in different body positions.

Client-Side A sensor positioned on the user collects gait data. From this data, a feature vector is generated. The features are extracted locally. As seen in Figure 4.1, the feature vector generated is characterized by the user u_i and by the body position b. An authentication request containing a user identifier, the body position and the feature vector is then sent over to the server.

Server-Side When an authentication request is received, the data first must pass through an input sanity check, in order to verify that it is real gait data. The goal of this first defensive layer, which is user-independent, is to reject malicious samples. If the feature vector received is classified as real gait, it is then submitted to a machine learning model. The model that authenticates the users is user-and-body-position-dependent, as shown in the figure. If the model finds a correspondence between the user and the data, the user is successfully authenticated, and a positive answer is sent back to the client. If the feature vector is rejected, the user is denied access.

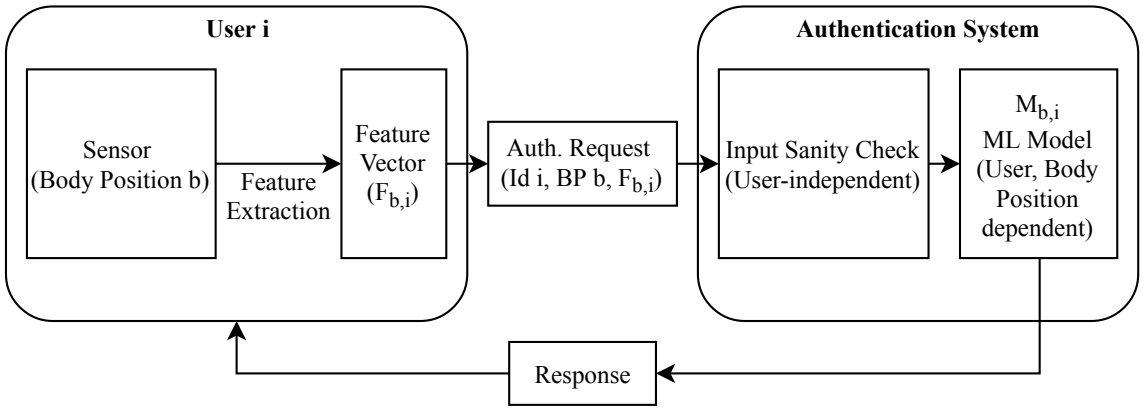


Figure 4.1: Authentication Process for user u_i. From a sensor in body position b, features $F_{b,i}$ are locally extracted. The authentication request $AR(i, b, F_{b,i})$containing the user identifier i, the body position b and the feature vector is submitted to the authentication system. The request must pass an input sanity check before being evaluated by the model $M_{b,i}$.

4.2 Input Sanity Check

In this section we explain why our system needs a first defensive layer and how we implemented it.

In a real system we expect security measures to be put in place to ensure that the data about to be processed by the system has not been tampered with. In our scenario, features are extracted locally on the user's device, therefore it is harder for the authentication server to verify that there was no external influence, such as adversarial modifications, on the authentication request. Adversarial samples, while evading the authentication model, are likely to present some non-natural features. Therefore, we want a security measure capable of distinguishing real gait data in order to reject adversarial samples before they are submitted to the model.

4.2.1 Implementation

We considered different possibilities to implement the input sanity check. The first solution we evaluated was to perform statistical tests. This would require to characterize the adversarial examples with statistical metrics like Maximum Mean Discrepancy (MMD) and Energy Distance (ED) [12]. However, this approach relies on having a set of adversarial examples to compute the metrics on and requires having batches of samples to be compared at test time. These two factors are incompatible with our scenario, since our system works on a sample by sample basis. The requirement of a set of adversarial examples also made us exclude a similar approach where

we would try to directly identify the malicious samples: in fact, even if we managed to obtain this set, the mechanism would have still been vulnerable to other kind of adversarial examples, leading to the so called "arms race" between the defender and the adversary, i.e. the vulnerability against new attack designs.

The final choice is a One-Class-SVM. Such classifiers have been used in the past to do anomaly detection, which is exactly what we need [23]. The goal of the system is to recognize real gait data, i.e. the same data used to train and test the one-sensor classifier and reject all the other samples. We want the sanity check to be user-independent therefore we use data from all combinations of user and body position to train and test the model. We train 9 models, one for each body position. We considered the possibility of a single, body-position-independent classifier but the performance was not good enough. While the average performance seemed good, when testing we discovered that different body positions were admitted with very different rates, which is the reason why we discarded this approach. During the testing process, first we ensure that real samples are accepted by the system, then we also need to test how well the model rejects non-gait samples. To do so, we submit to the model another set which represents possible outliers. The outliers set is constructed with data collected from users performing different activities, from all body positions.

4.3 Gait Authenticator

The gait-authentication system is an adaptation of the classifier used by Van hamme et al. [22].

4.3.1 Training Data

To train and test the classifier, we use data from the REALDISP dataset [3]. While the subset features different activities, we use the walking data collected from all available body positions of different subjects. Walking is chosen because is is the activity with the largest number of samples available. Among the subjects in the dataset, we selected the ones with the highest number of samples available, which is (~23). From the gait data collected by the sensors positioned on the users, we extracted a common set of features. Examples of the extracted features are mean, standard deviation, energy of the signal, kurtosis, mean average derivation and average resultant vector. The set is "common" in the sense that it is independent of the body position, of the subject and of the activity. Each subject performed each activity twice with slightly different sensor positioning. This allowed the authors to obtain a set of ideal and self-placed samples. Using both also increases the number of samples per subject to ~46. The data is then split in intervals of 128 samples, which

at the 50 Hz sampling frequency of the sensors results in around 2.5s each. The resulting feature vector, which includes both time and frequency domain features, has length 224.

4.3.2 Classifier

The goal of the classifier is to recognize the gait data of a user a reject all the others, in order to be used in the authentication scenario. To this purpose, a model is trained for each of the study subjects and each of the body positions. The models are trained and tested on a set constructed with data belonging to the subject (50% of the set) and data belonging to other subjects, in equal proportion. All data comes from the same sensor (body position). In each of these sets, data of a single user is labeled as positive while the data from the other subjects is labeled as negative. Every model is finally validated using n-fold cross-validation, each time leaving a different chunk out for testing, and accumulating false positives, false negatives, true positives and true negatives over the iterations. We repeated this process for every user in the dataset. To choose the learning algorithm we compared the average EER (Equal Error Rate) of the models for all body positions. Among the ensemble methods we tried, Random Forests and Gradient-Boosted Trees performed the best, followed by AdaBoosting and Bagging Trees. On the contrary, Support Vector Machines did not perform well due to the small amount of samples compared to the length of the feature vector. We ended up choosing Gradient-Boosted Trees because of the robustness w.r.t outliers, the ability to handle heterogeneous features and the resilience to overfitting.

4.3.3 Dimensionality Reduction

Having a classifier with reduced dimensionality provides some advantages. In tree-based classifiers some features have much more impact on classification than others, it is therefore reasonable to assume that those same features will have the most impact also on misclassification. Crafting an adversarial example against a tree classifier requires finding the optimal feature to modify and change it until another feature takes its place as optimal feature or until a maximum change is reached. Working with a reduced set of features makes this task much easier, especially in iteration-based methods, where the complexity of the algorithm is proportional to the dimension of the feature space. From now on the models we refer to will always be the reduced ones, since they provide a very good approximation of the original model.

4.3.4 Normalization

When deploying attacks against machine learning models we want our adversarial examples to be as hard to detect as possible. One way of decreasing the detectability of such samples is to make sure that the distance from the original samples is as small as possible. However, to obtain meaningful information from computing distances, the data must be normalized such that each feature is comparable. We apply a normalization algorithm that standardizes the features by removing the mean and scaling them to unit variance. The transformation is first applied on the training set, then the same transformation is applied on the test set. The transformations are applied in this order because test data should not influence the training data, since it is supposed to represent future data.

Chapter 5

Attack Methodology

In this chapter we introduce the methodology of our attack. We start by discussing the possible scenarios in which our system can be attacked, and what are the implications for the adversary in each of them. Then, we define the adversary model, stating his goals, knowledge and capabilities. Finally, we propose an algorithm for generating adversarial instances and we define how we evaluate their quality.

5.1 Attack Scenarios

In this section we describe what are the possible weak spots of the system that may cause an attacker to gain access to it.

Being reliant on a single sensor placed on the users, the system is vulnerable to observation attacks. An observation attack is an attack where the adversary obtains a trace of the user's gait data. For example, this could be achieved by placing a hidden sensor on him. If the adversary is able to place his sensor on the body position that the user is using for authentication, he is then capable of generating feature vectors resembling those of the user and fool the verification process. Being generated from real gait data, these vectors are also likely to be admitted by the input sanity check. The attacker could also obtain user's data from his smartphone. As an example, the HTML5 API provides means of obtaining accelerometer and gyroscope data, which, while not being the exact data used in the system, could be used to generate realistic features. The adversary could also obtain the trace of gait data directly from the user's sensor. With the sensors being connected to the Internet and possibly integrated in a smartphone, it is not unreasonable to assume that a sensor can be compromised and leak data.

However, the system is not effective also if the adversary is able to recreate the user traces on his own or if he can craft ad-hoc samples that evade the machine learning models. The second case, i.e. "evasion attacks", is the subject of our study.

Evasion attacks exploit a weakness of machine learning models: for virtually any instance x correctly classified by the model, it is possible to find a small perturbation δ such that $x+\delta$ is misclassified. The magnitude of the perturbation needed to make an instance evade the model often depends on how close this instance is to being misclassified. The instances on which we apply the adversarial perturbations are the starting points of our attack. We study two scenarios where different data is available to the adversary, and therefore the starting points are chosen differently. In the first scenario the adversary has access only to his own data, while in the second scenario also user data is available.

5.1.1 Attack Scenario: Adversary Data Adaptation

In the first scenario we analyze the system when no sensor is compromised. The goal of the adversary to authenticate as another user. To do so, he tries to generate samples that evade the model associated with the sensor the target user is using. To generate adversarial examples, he needs a good starting point. Since the adversary does not have access to any data other than his own, he uses his gait data as starting samples. We assume that the adversary is himself a user of the system, therefore, he knows the features that are computed by his sensors. We also assume that he knows the body position where the user's sensor is placed. This allows the adversary to choose a good starting point by positioning his sensor in the same body positions of the victim. Finally, the adversary sends an authentication request constructed with the user identifier and the crafted feature vector. The process is shown in Figure 5.1: the feature vector $F_{b,A}$, belonging to the adversary A and obtained from position b, is modified and sent to the server together with the identifier of user u_i. If it passes the sanity check, the sample is submitted to the model $M_{b,i}$. If the input sanity check and the user-and-body-position-dependent model are evaded, the adversary successfully logs in as the victim.

5.1.2 Attack Scenario: User Data Adaptation

In this second attack scenario, we analyze the effect of a compromised sensor leaking data to the adversary. Clearly, in this scenario the adversary could just use the user data to authenticate, performing what would be an observation attack. However, other studies proposed systems similar to ours that employ more than one sensor [22], precisely with the aim of preventing observation attacks. In such systems, the additional sensors prevent a single data leak from compromising the entire system. For this reason, it is interesting to evaluate whether having access to a trace from a body position is a threat to the security of a sensor placed in another body position. Figure 5.2 shows an attack in this second scenario. The adversary modifies a feature vector of user u_i, obtained from a compromised sensor in position b_1. When the

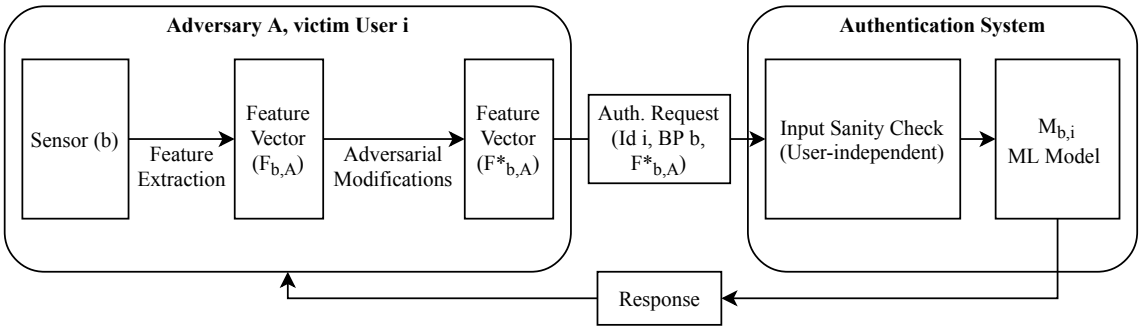


Figure 5.1: Evasion with adversary data as starting point. The feature vector obtained from a sensor placed on the adversary A in position b is modified to obtain an adversarial instance $F*_{b,A}$. The sample is attached to an authentication request with identifier i of user u_i (victim), which is then submitted to model $M_{b,i}$. If the attack is successful $F*_{b,A}$ is classified like a genuine sample $F_{b,i}$.

authentication request is sent to the authentication system, if it passes the sanity check it is then submitted to the model $M_{b_2,i}$. If also the model is successfully evaded, the adversary gains access to the system.

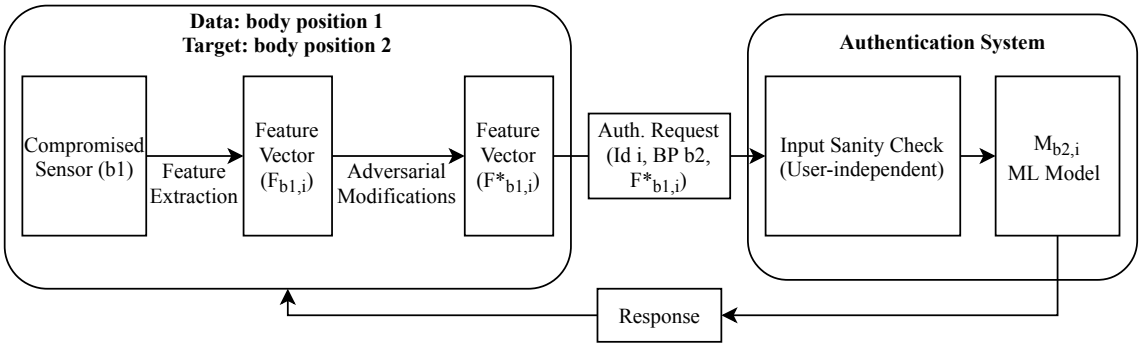


Figure 5.2: Evasion with user data as starting point. The adversarial instance is crafted modifying $F_{b_1,i}$, which is a genuine sample of user u_i collected from body position b_1. The feature vector $F*_{b_1,i}$ is attached to an authentication request for user u_i and body position b_2. If the attack is successful $F*_{b_1,i}$ is recognized as $F_{b_2,i}$ and the adversary gains access to the system.

The substantial difference between the two scenario are the options for choosing the starting point of his attack. The goal of studying different scenarios is to evaluate which of the two provides a better starting point for generating adversarial samples. A starting point is "better" if adversarial samples generated from it evade the model and they are harder to detect. We will cover the evaluation of an attack's performance in more detail in the last part of this chapter.

5.2 Modeling the adversary

In this section we model the adversary to define his goals, his knowledge of the system, and his capabilities.

Goals The goal of the adversary is to obtain evading instances that are also undetected by the sanity check. To do so, he applies some modification on real data, the starting point of his attack. The objective is to find which are the best starting points that allow the adversary to obtain misclassified instances with the smallest perturbations, and, at the same time, have high evasion rates against the sanity check.

Knowledge We described what data the adversary has access to in the previous section. We assume that he is a user of the system, therefore he knows the feature space of the gait authenticator, since it is the same for every user. He can obtain the feature vectors extracted from his own data, since they are locally computed on his sensor. Finally, we assume that he can get feedback from the decision function of any individual model $M_{b,i}$ by submitting an authentication request with identifier i. This means that he can submit a feature vector and in return obtain a real value representing the decision of the system. He knows about the presence of an input sanity check but he does not have insight on its working principle.

Capabilities The adversary does not have any way of influencing the model, i.e. the internal workings of the authenticator are fixed. However, there are no constraint regarding how the feature vectors are generated, therefore, the adversary is free to choose the starting samples and the magnitude of the modifications. However, he must consider that larger modifications might make the adversarial sample easier to detect.

5.3 Attacking Ensemble Learners

Here we explain our proposed algorithm for generating adversarial instances. There are multiple possible approaches to the attack, some of which have been studied in the literature by Kantchelian et al. [15] and by Grosse et al. [12]. The biggest drawback of the proposed solutions is that they require a perfect knowledge of the model to be attacked, while our solution requires a black-box of the decision function. On the other side, our solution is slower, especially for very high-dimensional feature spaces. However, for our study case, speed is not a priority, therefore we chose the solution that requires less knowledge from the adversary's part. The algorithm is based on an iterative approach.

Let M be the decision function of the model. Starting from a sample x, assume $M(x) < 0$. At each iteration we want to compute the best change of a single dimension of x such that $M(x')$ increases. The algorithm stops when $M(x') > 0$ or when the modification budget is exceeded. The most challenging part is finding both the best step and the best dimension to change at the same time. Therefore, we search the best dimension for a fixed step a and, if a solution is not found, we repeat the process with a new value for the step size. This approach also allows us to easily impose a budget on the maximum modification by limiting the maximum step size.
At each iteration, the problem we want to solve is:

$$\text{maximize } M(x' + \alpha u) \text{ subject to } u_i = 1 \text{ and } \forall j \neq i, u_j = 0$$

We compute $M(x' + au)$ for all dimensions and store the best dimension for the current step size. As long as $M(x')$ keeps increasing we repeat the process without changing the step size. For very small step sizes, it is possible that $M(x')$ does not increase at all. If this happens, the step increase or changes sign. We test positive and negatives steps in order to explore all the possible directions.

5.3.1 Performance evaluation

The step size limit imposes a budget on the maximum modification allowed for a single feature. It is also interesting to analyze the distance from the original sample in terms of the entire sample. To do so, we have different options [15]:

- The Hamming distance $\sum_{i=1}^{n} a_{x_i \neq x'_i}$, corresponding to the number of features modified, encourages the most localized adversarial deformations without constraining their magnitude.
- The L_1 distance $\sum_{i=1}^{n} |x_i - x'_i|$ controls the magnitude of adversarial deformations, but does not constrain the localization.
- The L_2 distance $\sqrt{\sum_{i=1}^{n}(x_i - x'_i)^2}$ penalizes large deformations and therefore encourages less localized but small adversarial deformations.
- The L_∞ distance $\max_i |x_i - x'_i|$ encourages uniformly spread adversarial deformations with the smallest possible magnitude.

For different maximum step sizes, we compute the distance from the original samples to see, on average, how much adversarial samples evading the model differ from the real data. The distance chosen is the L_2 distance because we assume that big, localized changes have the highest cost in terms of detectability of the adversarial sample. Finally, all the generated adversarial samples are tested against

the sanity check model to see how many of the samples which evaded the individual model are also undetected by the general check. We expect that samples generated with more lax constraints, i.e. those more distant from the original, to have worse evasion rates against the sanity check.

Chapter 6

Evaluation

In this chapter we evaluate the effectiveness of the approach described in the previous two chapters. First, we introduce the dataset that we use both to test the system and to conduct the experiments. We start by testing and validating the gait authenticator and the input sanity check. The rest of the chapter is structured in the following way: for each experiment we describe the reasoning behind it, what data we use, how we implement it and finally discuss the results. In the first experiment we analyze these aspects for the samples generated with adversary data, while in the second one we analyze those generated with user data, focusing on the differences between the two.

6.1 Dataset

The data used in the experiments is taken from the REALDISP (REAListic sensor DISPlacement) dataset, which is a publicly available benchmark dataset collected by Banos et al. [3]. The dataset contains gait data of multiple users, collected while performing different activities.

The dataset features 17 subjects and it's sampled at 50 Hz, which is deemed to be sufficient for the 33 considered activities. 9 inertial measurements units are placed on the subjects, in the following body positions: left calf (LC), left thigh (LT), right calf (RC), right thigh (RT), back (BACK), left lower arm (LLA), left upper arm (LUA), right lower arm (RLA), right upper arm (RUA). Each unit can sense acceleration, rate of turn, magnetic field and can derive the orientation estimates of the sensor frame with respect to the Earth reference [3]. 3D accelerometer, 3D gyroscope, 3D magnetic field orientation and 4D quaternions data is collected. The same set of activities is performed in two scenarios: first, the "ideal-placement" scenario, where the sensors are positioned by the authors in the optimal position, and second, the "self-placement" scenario, where the users are asked to position the

sensors themselves, without additional hints from the instructor. This last scenario tries to simulate the variability that may happen in a daily usage of the sensors.

While originally collected for activity recognition studies, the dataset works well for authentication and identification purposes. In this work, we use data from all body positions and from four activities: walking, running, jogging and cycling. In all cases, a combination of ideally-placed and self-placed data is used, in order to provide the best approximation of a real-life application.

6.2 Performance Evaluation of the Authentication System

6.2.1 Input Sanity Check

The goal of this model is to accept only data classified as "real gait" and reject all the other samples. We train a user-independent model for each body position. In the first iteration, the system was also body-position-independent. However, if tested against samples from different body positions, the performance would be too different to be accepted. Some positions would have almost perfect accuracy, while others would have 30% error rate. The data used is walking data, collected from all the nine available body positions of the same subjects used to train the gait authenticator. The models are validated with 4-fold cross validation, where we train the model with 75% of the data and leave 25% out for testing and repeat the process four times with different splits. The data is shuffled to avoid having some users appear only of the two sets. We use walking data from all users to test the acceptance rate, while we use data from other activities (running, jogging, cycling) to build an outliers set and test the rejection rate.

Performance

We obtained the best performance with a One Class SVM with RBF kernel, and parameters nu=0.05, gamma=0.015. To evaluate the performance, we compute the FAR and the FRR obtained after testing the system with the test set, left out of the training phase, and the outliers set. The results are shown in table 6.1.

The system performs well, with average False Rejection Rate smaller than ~6% and average False Acceptance Rate smaller than ~9%. The only outlier seems to be the back data, which presents a FAR higher than average at ~20%. A possible intuitive explanation for this behavior could be that the back movement is similar for different activities.

Body Position	FRR	FAR
RLA	0.061	0.086
RUA	0.061	0.017
BACK	0.042	0.202
LUA	0.061	0.079
LLA	0.047	0.057
RC	0.051	0.083
LC	0.061	0.008
RT	0.056	0.089
LT	0.042	0.075

Table 6.1: Comparison of FRR (Positive samples rejected) and FAR (Negative samples accepted) for different body positions.

6.2.2 Gait Authenticator

To train and test the classifier, we use walking data from all available body positions of different subjects. Walking is the activity with the largest number of samples available, other than being the most common activity. We train a model $M_{i,b}$ for each user u_i and body position b. The positive class is constructed with data belonging to the subject, while the negative class is an equal distribution of samples from other subjects. We validate every model with n-fold cross-validation, accumulating false positives, false negatives, true positives and true negatives over the iterations, from which we compute the ROC curve. We test all the three models (original, dimensionality-reduced, and reduced-normalized) with the same method.

Performance

To evaluate the performance of the classifier we compute FAR (False Acceptance Rate) and FRR (False Rejection Rate). We want the two rates to be as low as possible. An approach to find a compromise is the EER (Equal Error Rate), which indicates that the FAR and the FRR are equal. The lowest the EER, the higher the accuracy of the system. As shown in table 6.2 there is almost no loss in performance after applying dimensionality reduction and normalization.

Body Position	Original	Reduced	Reduced-Normalized
RLA	0.101	0.089	0.097
RUA	0.078	0.087	0.070
BACK	0.043	0.074	0.065
LUA	0.078	0.089	0.087
LLA	0.087	0.093	0.080
RC	0.044	0.052	0.063
LC	0.057	0.043	0.044
RT	0.047	0.056	0.044
LT	0.072	0.052	0.052

Table 6.2: Comparison of the EERs of the three models.

6.3 Evasion Attack: Adapting Adversary Data

The goal of these experiments is to evaluate the resilience of the system to evasion attacks. If the adversary can obtain a trace of the user gait data, the system is vulnerable to observation attacks. In this experiment we evaluate the evasion abilities of an adversary who has access only to his own data.

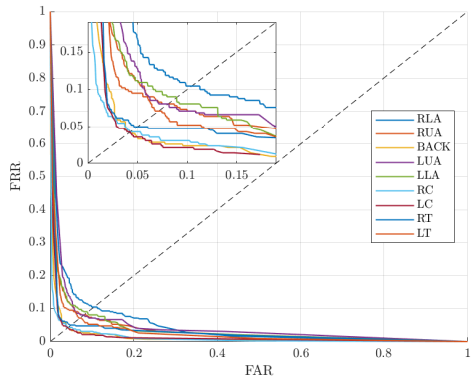
Assumptions To evaluate this scenario, we assume that the adversary is a user of the system. As user of the system, he knows:

- What data is collected by the sensors and what are the possible body positions in which they can be placed.
- The feature space of the machine learning models.
- The user identifier of his victim, to be inserted in the authentication request. This can be an e-mail address or a numeric identifier.

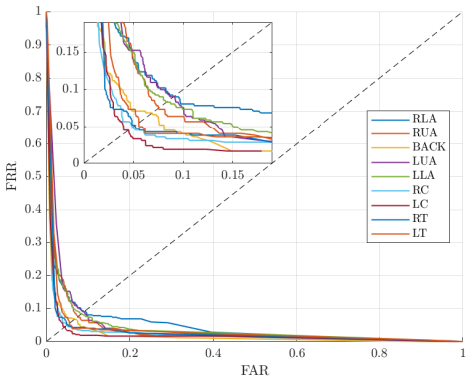
Since the feature vectors are computed locally, we assume that he has access to them and can modify them. Finally, we assume that he can probe the model $M_{i,b}$ specific to user u_i and body position b and retrieve the output of the decision function.

6.3.1 Generating Adversarial Samples

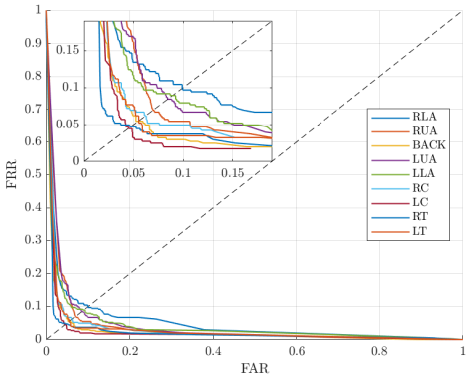
The goal of the adversary is to access the system as user u_i which is using a sensor placed on body position b. To do so, he crafts adversarial samples using the algorithm explained in the previous chapter. Generating adversarial samples requires a starting point. Since the only data available to the adversary is his own, we use data



(a) Original Model



(b) Reduced Model



(c) Reduced-Normalized Model

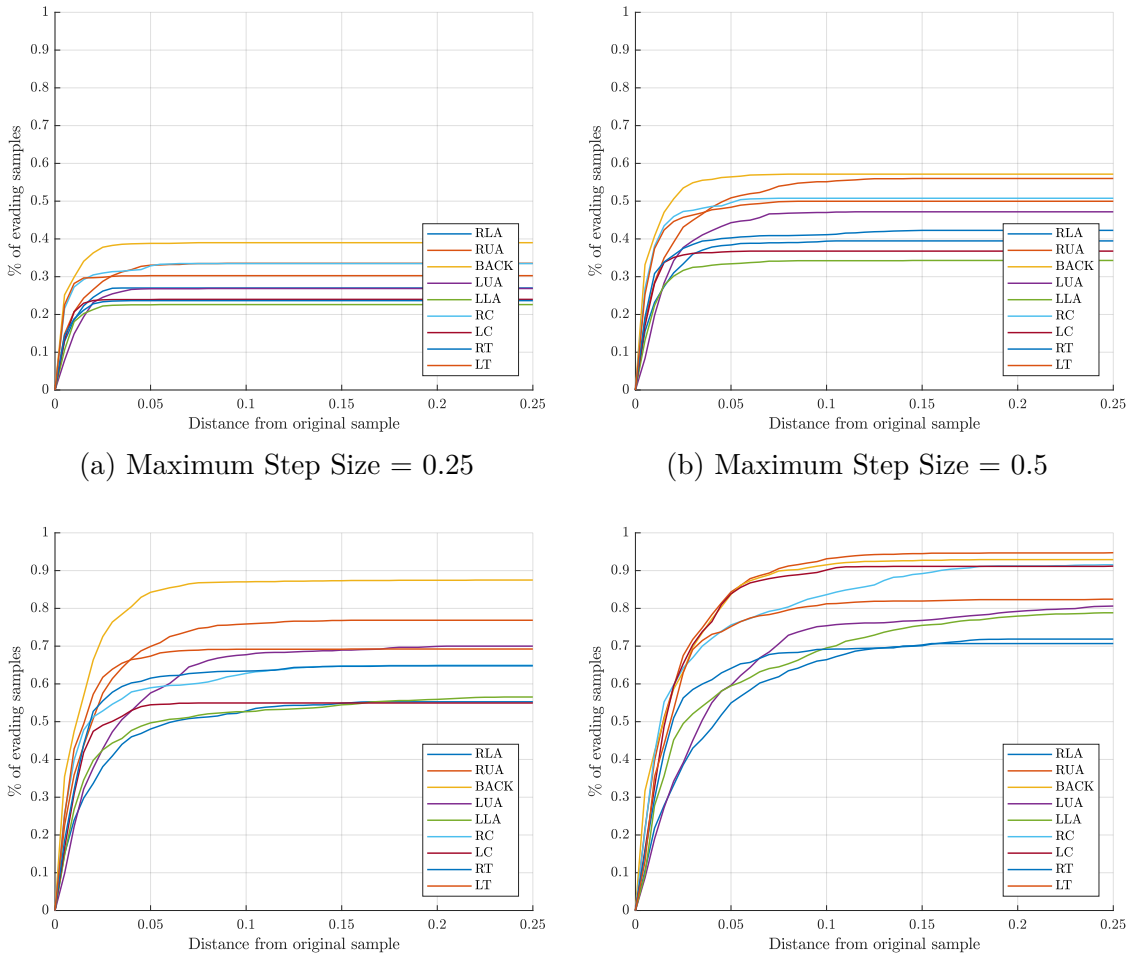
Figure 6.1: Comparison of the ROC curves of the three models. The three graphs being very similar show that the three models have very similar performance.

from a sensor placed in the same body position that the victim is using. To test this scenario we take a user u_i out of the set of users and consider him the adversary. All the users u_j, $j \neq i$ are considered victims. We repeat this experiment for all the possible combinations of adversary, victims and body positions.

$$\text{Adversary: } u_{i,b}, \text{ Victim: } u_{j,b} \; \forall j \neq i$$

We generate the adversarial samples with different values for maximum step size $s = \{0.25, 0.5, 1, 2\}$ and we compute the distance of the generated samples from the original ones.

In Figure 6.2 we compare the effect of the different step sizes on the evasion effectiveness. The limit on the maximum step size also acts as a limit on the maximum tolerated distance from the original sample, as we can see from the plateaus



(a) Maximum Step Size = 0.25

(b) Maximum Step Size = 0.5

(c) Maximum Step Size = 1

(d) Maximum Step Size = 2

Figure 6.2: Comparison of the performance for different step sizes and different targets (body positions). Each points shows, given a distance threshold, the percentage of samples evading the model, for each body position.

which are reached faster for small step sizes. The goal of the step size constraint is to limit the amount of modification on a single feature. Removing the constraint on the step size would lead to higher percentages of evasion, but we assume that the malicious samples would become easier to detect by a careful defender.

From this part of the experiment it is possible obtain an initial assessment of which are the best body positions to use for authentication purposes in this system. The body positions present an overall similar behavior, with the back data consistently being the most vulnerable, while the lower arms data, both left and right, seems to be the most resilient. The higher vulnerability of back sensors may be

due to the back movement being less characterizing. This results also suggest that integrating the sensors in a smart-watch or a similar device may be the best option.

Step Size	$d < 0.01$	$d < 0.02$	$d < 0.05$	$d < 0.10$	$d < 0.25$
0.25	0.219	0.265	0.288	0.289	0.289
0.50	0.299	0.385	0.443	0.457	0.460
1.00	0.335	0.485	0.613	0.652	0.667
2.00	0.321	0.504	0.712	0.800	0.839

Table 6.3: Percentage of evading samples by distance and step-size. Average of all body positions.

In Table 6.3 we compare the average performance of the attack on every body position, for each step size. The modifications needed to obtain an evading sample are often very small: even with the stricter constraints ($d < 0.01, s = 0.25$) ~22% of the samples are successfully modified. Relaxing the constraints ($d < 0.05, s = 1$), over 60% of the samples can be modified to evade the model. This shows the vulnerability of the system and the necessity for additional checks to detect such samples.

6.3.2 Input Sanity Check

To address the vulnerability to adversarial samples, an input sanity check is deployed. The goal of this machine learning model is to detect malicious instances before they are submitted to the user-specific model. The adversarial samples we generated in the first part of the experiment must pass the sanity check in order to be effective. We also want to evaluate whether our assumptions regarding the correlation between step-size (distance) and detectability are true. First, for each body position, we submit the relative adversarial samples to the sanity check model to test the overall detectability of the generated samples. Finally, we evaluate the detectability with respect to the distance from the original samples.

Detectability vs Maximum Step-Size

As described in the previous chapter, this model is trained with data from all users of the system, from all body positions. The model is trained with 75% of the dataset, leaving 25% out for testing and repeating this process four times with different splits. Since the starting points we choose to generate the adversarial samples also belong to this dataset, at each iteration we test against the model only those generated from the data in the test set, in order to avoid biased results. The results are shown in Figure 6.3.

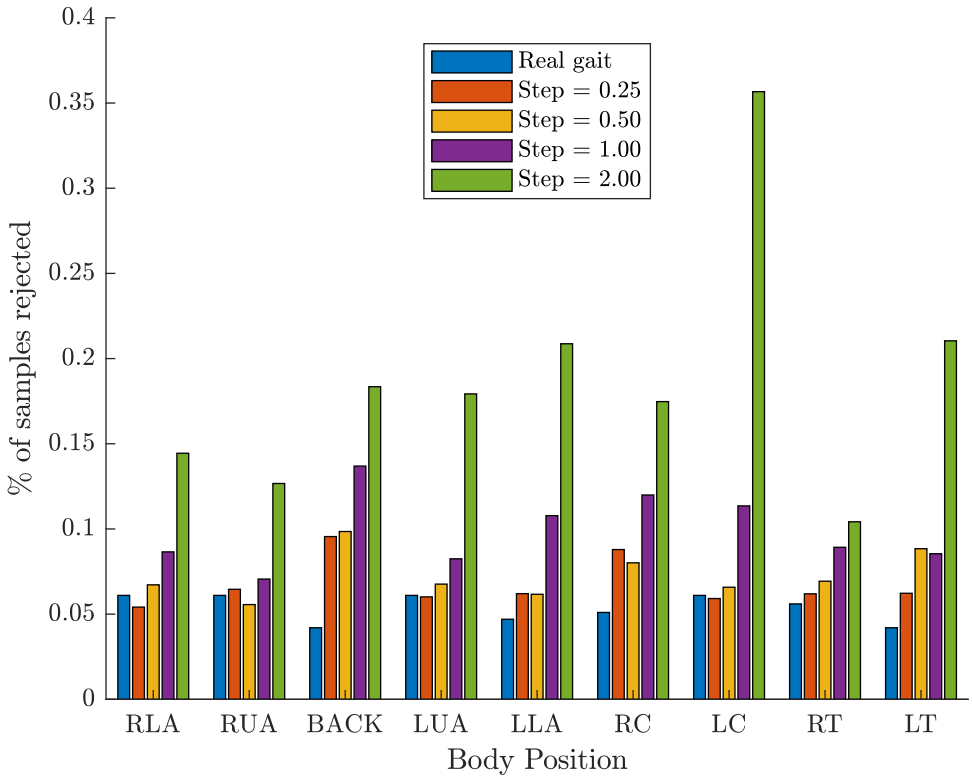


Figure 6.3: Adversarial samples rejected by body position and maximum step size. Comparison with real gait data.

Different body positions show similar trends: the samples generated limiting the step size ($s = \{0.25, 0.5\}$) tend to be very close to the real samples in terms of rejection rate. The only exception is the back samples, where the difference with respect to real data is substantial even for small modifications. Starting from step size $s = 1$, there is a steep increase in the number of rejected samples, which becomes even more noticeable for $s = 2$. To read this data, we must take into account the number of samples in each dataset. In fact, there is a trade-off between how many samples can be generated and their effectiveness. With $s = 0.25$, less than 30% of the total samples can be successfully modified to obtain an evading one. However, these samples have a rejection rate almost equal to real data. On the other hand, with $s = 1$, over 60% of the samples can produce adversarial samples, but the rejection rate is twice as high as real data.

Detectability vs Distance

In the first section of this experiment we assumed that a larger distance from the original samples would cause the adversarial samples to be easier to detect. The analysis of the effect of the maximum step size is closely related to this statement, since allowing larger modifications on single features leads to larger distances. To test whether our assumption holds, we evaluate the average distance of the samples evading the model versus the average distance of the samples rejected by the system. It's important to notice that this evaluation is not about finding how many samples evade the sanity check but rather evaluating the differences between rejected and accepted samples. The results are shown in Figure 6.4. The experimental results seem to confirm that there is a relation between distance and detectability. The rejected samples present higher distance from the original ones (i.e. higher deformation). This is true for every body position.

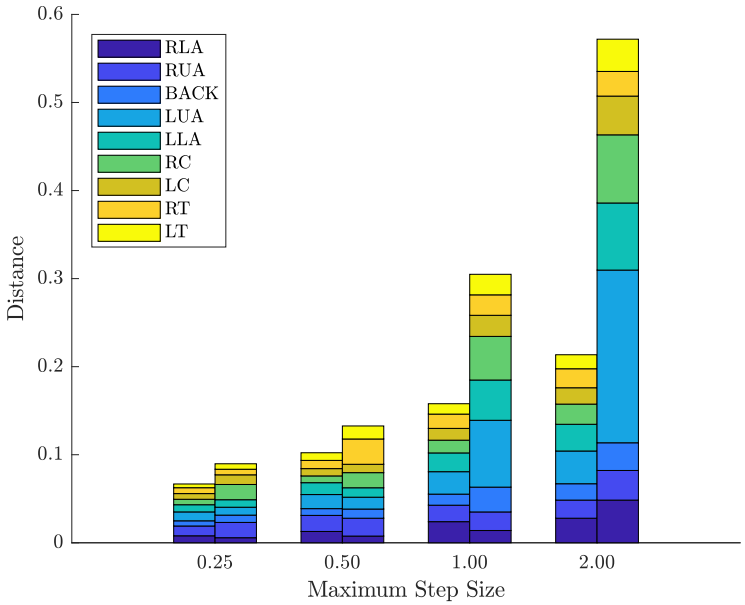


Figure 6.4: Average distance of evading samples (left) vs non-evading samples (right).

6.4 Evasion Attack: Adapting User Data

In Chapter 5 we talked about the implications of having a sensor compromised. While in this scenario the adversary could use this data to authenticate, there are a few reasons to study it. First, instead of a sensor being compromised, the adversary may obtain a user's gait data by placing a sensor on him. This sensor would likely not be in the same position that the user uses to authenticate. The second reason is the one explained earlier: authentication systems may rely on more than one sensor to verify the user. If we are able to generate adversarial instances from the compromised sensor, the effectiveness of the multi-sensor approach decreases greatly.

The goal of the experiment is to evaluate whether having different data available produces better adversarial samples and, if true, what are the best starting point for generating them. In other words, we want to test if having data from a compromised sensor in body position b_1 increases the vulnerability of a sensor in body position b_2. Throughout this evaluation, we focus on how the differences with respect to the first experiment.

Assumptions The assumptions regarding the adversary knowledge of the system (body positions, feature space, user identifiers) are the same as in the previous experiment. Additionally, we assume that the attacker has access to the trace of a sensor in a body position different from the one he wants to attack.

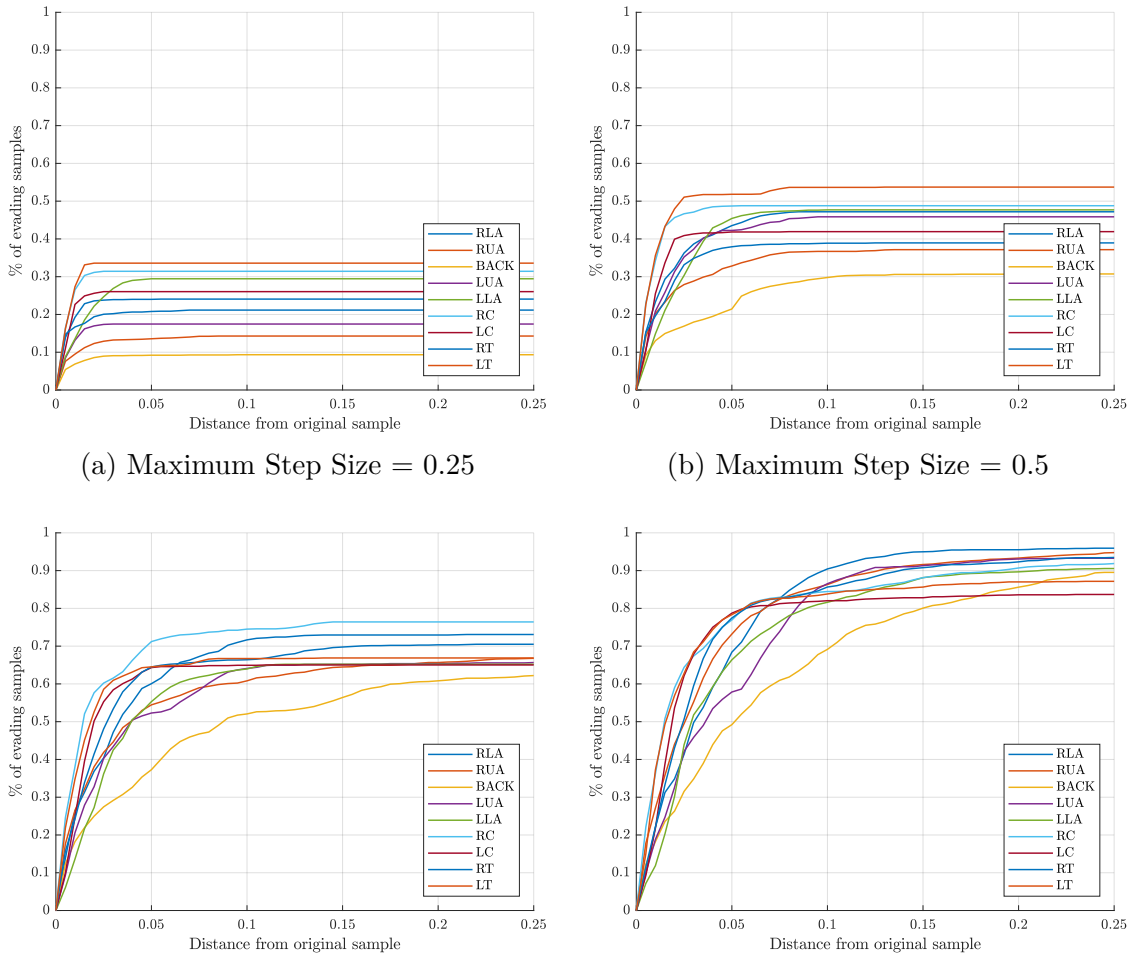
6.4.1 Generating Adversarial Samples

The goal of the adversary is to access the system as user u_i which is using a sensor placed on body positions b_j. The adversarial samples are generated using the same algorithm used in the previous experiment. However, for the reason listed earlier, the adversary has access to a trace of the same user u_i collected from body position b_k. We test this scenario taking user u_i as victim, and using data from his own sensor $b_{i,j}$ to generate adversarial samples targeting the model M_{i,b_k}, for all combinations of body positions.

$$\text{Victim: } u_i, \text{ Target sensor: } b_j, \text{ Data available (user data): } b_k \ \forall k \neq j$$

We generate the adversarial samples with different values for maximum step size $s = \{0.25, 0.5, 1, 2\}$ and we compute the distance of the generated samples from the original ones.

In Figure 6.5 we compare the effect of the different step sizes on the evasion effectiveness. The trend is similar to the one observed in the first experiment, but there are some significant differences. First, for small step size $s = 0.25$ the percentage of samples evaded is much lower than in the first experiment. This suggests that same



(a) Maximum Step Size = 0.25

(b) Maximum Step Size = 0.5

(c) Maximum Step Size = 1

(d) Maximum Step Size = 2

Figure 6.5: Comparison of the performance for different step sizes and different targets (body positions). Each points shows, given a distance threshold, the percentage of samples evading the model, for each body position.

subject, different position starting points may need larger modifications with respect to different subject (adversary), same position starting points. Increasing the step size, we have an increase in evasion rate similar to the first experiment, for all body positions. However, the steep increase in the graphs shows that the distance from the original samples tends to be smaller, suggesting that such samples either evade the model with relatively small modifications or cannot evade it at all. Regarding body positions, the evasion rates are very similar for all except the back, which seems harder to evade for all step sizes. This is the opposite of what we found in the first experiment, where the back sensors were the most vulnerable. However, the

difference in starting points can explain this result: while using same position data as starting point makes back sensors easier to evade due to back movement being less unique, using different position data as starting point makes evasion of the back sensors harder because the movement patterns are more different with respect to all other body positions. This also suggests that there is a correlation between the data of other body positions.

Step Size	d<0.01	d<0.02	d<0.05	d<0.10	d<0.25
0.25	0.173	0.215	0.228	0.230	0.230
0.50	0.230	0.327	0.406	0.434	0.436
1.00	0.252	0.402	0.582	0.650	0.680
2.00	0.269	0.423	0.696	0.833	0.911

Table 6.4: Percentage of evading samples by distance and step-size.

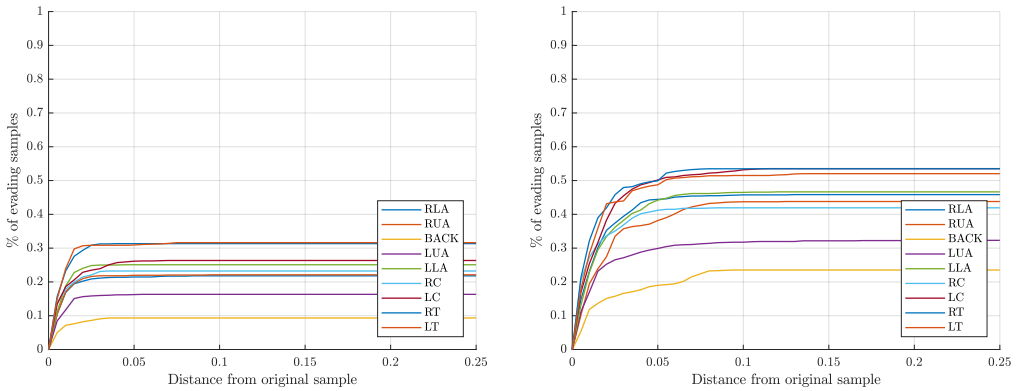
In Table 6.4 we compare the average performance of the attack on every body position, for each step size. The evasion rates by step size and distance are in general smaller with respect to the first experiment. However, the performance seems to scale better with respect to the step size, again suggesting that larger modifications are needed to generate evading samples.

Starting Positions

Until now we studied which sensors are harder or easier to evade. For this second scenario it is also interesting to study which body positions are better to be used as starting point for evading other positions. While in the first part we evaluated which sensors are better to have in the system, as they are harder to evade, in this second part we evaluate which sensors is better not to have, as they are more suited for evading others. For every body position, we show how many of the samples generated starting from it evade the model. The results are shown in Figure 6.6.

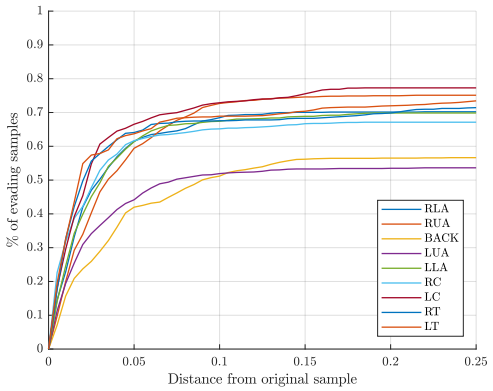
Most starting points present a similar overall behavior, again with the exception of the back, which seems to be less effective for generating evading samples for other positions. Having one sensor placed on the back of the users seems to be the best option in this second scenario, since back sensors result the hardest to evade and also the worst for generating adversarial samples.

From the study done until now, samples generated from user data seem to have worse performance with respect to the ones generated from adversary data. The starting points provided by the adversary data (samples from the same body position of the target) seem to be better than samples obtained from a different body position of the victim. However, to confirm this hypothesis, we first need to check the performance of these adversarial samples against the input sanity check.

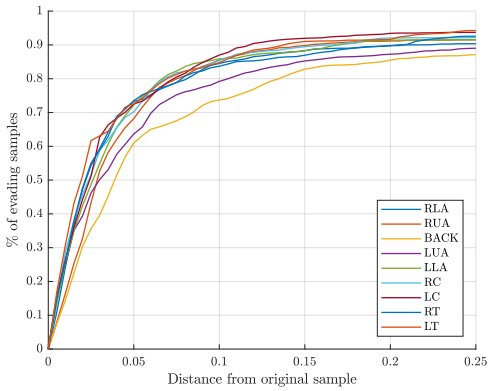


(a) Maximum Step Size = 0.25

(b) Maximum Step Size = 0.5



(c) Maximum Step Size = 1



(d) Maximum Step Size = 2

Figure 6.6: Comparison of the performance of different starting points (body positions) and different step sizes. Each points shows, given a distance threshold, the percentage of samples generated from that body position that evade the model.

6.4.2 Input Sanity Check

The adversarial samples we generated in the first part of the experiment must pass the sanity check in order to be effective. In this section we evaluate the performance of the adversarial samples generated from user data and we compare it to the performance of the samples generated in the first experiment. As in the first experiment, we test against the model only the samples generated from data in the test set. The results are shown in Figure 6.7.

The performance of the new adversarial samples against the sanity check is much lower than the first experiment. While in the first experiment we had evasion rates very similar to real gait data, with a significant decrease in performance only for large step sizes, now most of the samples are rejected by the sanity check, even those

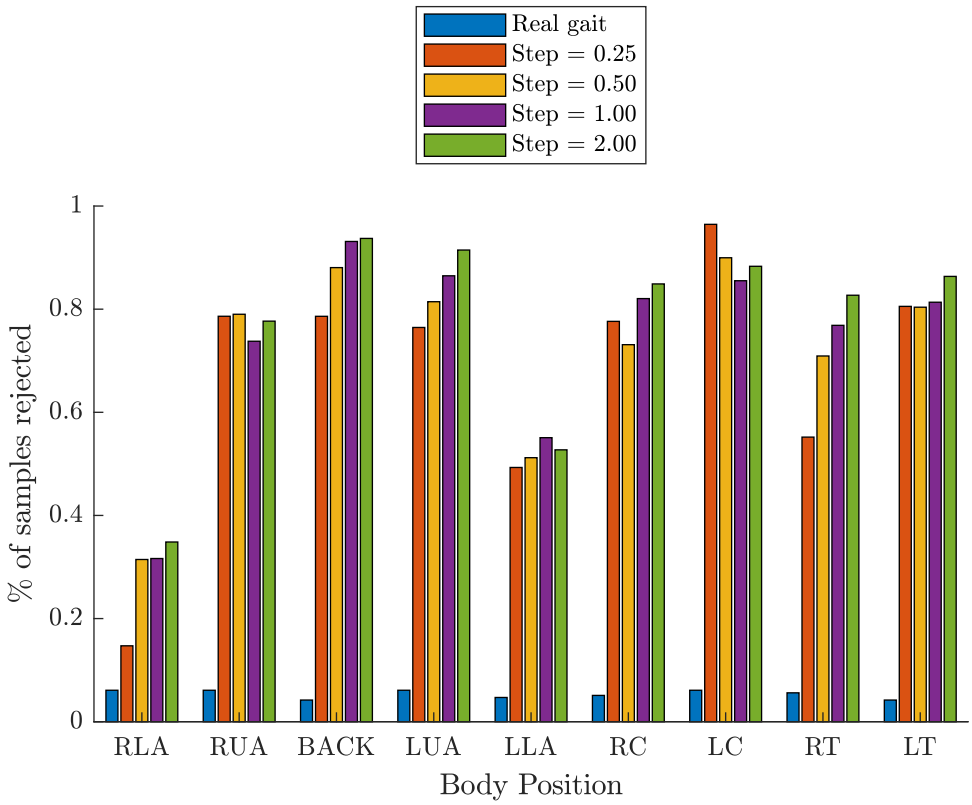


Figure 6.7: Adversarial samples rejected by body position and maximum step size. Comparison with real gait data.

generated with very low step size. The trend of samples generated with large step size being easier to detect also shows up in these results, albeit it is less evident. The higher rejection rates can be explained by the nature of the starting points: starting from body positions different from the one we want to attack causes the samples to preserve some features typical of the original body position. Therefore, since the sanity check model is position dependent, it is much easier for it to detect adversarial samples. These results support the hypothesis we made in the first part of experiment that samples generated from user data perform worse and in general pose a lesser threat to the system. We conclude that, while having a sensor compromised reduces the overall security of the system, it does not impact the security of other sensors that may be present. Also, if the adversary is able to obtain a trace from a body position different from the one used for authentication, the starting points will perform much worse than if he used his own data.

Chapter 7

Discussion

In this chapter we discuss the results of the experiments conducted in Chapter 6. First we present the conclusion that can be drawn from the work done and the contribution, then we discuss limitations and how can they be addressed in future work.

Results and Contribution The main goal of our work was to test the effectiveness of evasion attacks against ensemble models. We developed a novel attack to generate adversarial samples starting from genuine ones. The system on which we conducted the attack works very well for authentication purposes (EER between ~4% and ~9%) but was not tested against smart adversaries. The attack is quite effective and allowed us to gain some insight about the effect of different body positions in a gait-based authentication system. We concluded that sensors positioned on the back of the users are the most vulnerable to adversarial attacks and, if possible, their usage should be avoided. We studied the effect of overall deformation and single feature modification on the effectiveness of adversarial samples proving that our hypothesis regarding the relation between distance from the original samples and evasion rate was correct. Finally, we tested whether having access to the victim data from a body position different from the attacked one could provide better starting points for the evasion attack. This is important in presence of multi-sensor systems or if the adversary is capable of placing a sensor on the victim. We concluded that having access to this additional data does not provide an advantage for the adversary, as the new starting points this scenario offers perform worse than those already available.

Finally, we deployed a simple defensive mechanism based on outliers detection with the goal of rejecting adversarial samples before they are submitted to the authentication models. The system proved to be effective against samples generated from body positions different from the one attacked with high rejection rates of ~80%. However, it fell short when dealing with samples generated by an adversary

using sensors placed in the same position used by the victim, with rejection rates higher than real data but overall quite low at ~10%.

We showed that ensemble learners work well for authentication purposes but, without any additional defense, they are weak to evasion attacks. A smart adversary that can choose good starting points for his attack can obtain access to the system rather easily by crafting adversarial samples. The main weakness of such system seems to be the reliance on a specific body position; data of other subjects taken from the same body position on which the system is based on may be similar enough to genuine data to provide very good starting points. Samples generated in this way proved to have very high evasion rate when tested against our defensive system. Finally, studying what characterizes a good adversarial sample provides an insight on which additional measures can be taken to detect them.

Limitations and Future Work While some conclusions we drawn from our work can be generalized to real systems, some may be system specific and would need to be validated by repeating the experiments on a larger dataset featuring more users. The dataset we used is publicly available, which guarantees reproducibility of our research results, but it only features 17 subjects which may not be enough to generalize our results.

We proved that ensemble models are rather easy to evade, but we did so modeling an adversary with a very extensive knowledge of the system, which may be too much for a realistic scenario. A goal of future work should be to study the effectiveness of an attacker with reduced capabilities, for example by removing the possibility of directly acting on the features or by reducing the knowledge of the decision function. This would require the adversary to generate the gait data normally collected by the sensors, and doing so in a way that produces adversarial samples when features are extracted. Another option to be tested is the possibility of exploiting the intra-technique transferrability property of machine learning models defined by N. Papernot et al.[20], which states that an adversarial sample evading a model M_1 may also evade a model M_2 obtained with different initialization parameters or dataset. This approach would remove the need of the adversary of knowing the decision function of the attacked model.

The results we obtained regarding the effect of body positions seem consistent across the whole dataset, but they need validation from testing on larger datasets and possibly different systems, as they may be dataset or system dependent.

Finally, future work should focus on what defenses can be implemented to counter evasion attacks. The system we proposed shows some weakness related to the influence that body positions have on it. Defense mechanisms could be implemented by exploiting the characteristics of adversarial samples we described in the previous section. Statistical tests were not effective on our case study because it worked on

a sample by sample basis. However, in a scenario where features are extracted on server side and users sends gait data to the server, they may prove to be an effective way of rejecting malicious batches. Finally, instead of trying to reject samples before they are submitted to the system, it should be possible to make the machine learning models more resilient using adversarial training.

Chapter 8

Conclusion

Evaluating the resilience of an authentication system is a fundamental step of its deployment. We chose to work on tree ensembles because attacks against them were a relatively unexplored subject. Moreover, we are not aware of similar work in the context of behavioral authentication. We have shown that tree ensembles can be evaded rather easily. We proposed an iterative algorithm that can generate adversarial samples from any starting point, while constraining the single feature modification. We tested the approach on an adaptation of an existing authentication system, obtaining evasion rates of up to 80%. We compared the characteristics of the adversarial instances (modification, constraints at generation time) to their evasion rate, showing a trade-off between the number of samples that can be generated and their quality. We showed how evading instances are characterized by a much higher L_2 distance with respect to non-evading ones.

To have a complete view of the adversary capabilities, we repeated the experiments in different scenarios, with different data available to be modified. The choice of the starting point turned out to be the most important characteristic for obtaining a good evading sample. This showed that collecting data of similar characteristics to the target of the attack is the most important step to obtain a successful attack.

The results obtained by the attack expose a need for additional security checks. Our proposed solution tries to identify adversarial instances as outliers, in order to reject them before they are submitted to the tree ensemble. The defensive mechanism we deployed is based on a one-class SVM. The system showed varying levels of performance, depending on the starting points chosen by the attacker. In the first scenario, it was able to reject from to 10% to 40% of the samples, depending on the constraints at generation time. It proved very effective in the second scenario, showing that the approach has potential. However, the presence of very different performances depending on the scenario, shows that system specific notions must be taken into account very carefully when designing this kind of system. Outliers detection alone, while improving the resilience of the system, is not enough and

should be used in combination with other forms of protection.

The security evaluation shows that evasion attacks are a real threat that must be taken into account when designing behavioral authentication systems. Effective countermeasures need to be put in place for a system to be deployed. In fact, the relative easiness of generating adversarial instances can make the impact of evasion attacks very significant.

Bibliography

[1] A. Al Abdulwahid, N. Clarke, I. Stengel, S. Furnell, and C. Reich. Continuous and transparent multimodal authentication: reviewing the state of the art. *Cluster Computing*, 19(1):455–474, Mar 2016.

[2] A. Annadhorai, E. Guenterberg, J. Barnes, K. Haraga, and R. Jafari. Human identification by gait analysis. In *Proceedings of the 2Nd International Workshop on Systems and Networking Support for Health Care and Assisted Living Environments*, HealthNet '08, pages 11:1–11:3, New York, NY, USA, 2008. ACM.

[3] O. Baños, M. Damas, H. Pomares, I. Rojas, M. A. Tóth, and O. Amft. A benchmark dataset to evaluate sensor displacement in activity recognition. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, pages 1026–1035, New York, NY, USA, 2012. ACM.

[4] L. Bao and S. S. Intille. Activity recognition from user-annotated acceleration data. In A. Ferscha and F. Mattern, editors, *Pervasive Computing*, pages 1–17, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[5] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli. Evasion attacks against machine learning at test time. In H. Blockeel, K. Kersting, S. Nijssen, and F. Železný, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 387–402, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[6] B. Biggio, L. Didaci, G. Fumera, and F. Roli. Poisoning attacks to compromise face templates. In *6th IAPR Int'l Conf. on Biometrics (ICB)*, Madrid, Spain, 06/2013 2013.

[7] J. E. Cutting and L. T. Kozlowski. Recognizing friends by their walk: Gait perception without familiarity cues. *Bulletin of the Psychonomic Society*, 9(5):353–356, May 1977.

[8] M. O. Derawi, C. Nickel, P. Bours, and C. Busch. Unobtrusive user-authentication on mobile phones using biometric gait recognition. In *2010 Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 306–311, Oct 2010.

[9] R. D. Findling, M. Hölzl, and R. Mayrhofer. Mobile gait match-on-card authentication from acceleration data with offline-simplified models. In *Proceedings of the 14th International Conference on Advances in Mobile Computing and Multi Media*, MoMM '16, pages 250–260, New York, NY, USA, 2016. ACM.

[10] D. Gafurov, E. Snekkenes, and P. Bours. Gait authentication and identification using wearable accelerometer sensor. In *2007 IEEE Workshop on Automatic Identification Advanced Technologies*, pages 220–225, June 2007.

[11] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and Harnessing Adversarial Examples. *ArXiv e-prints*, Dec. 2014.

[12] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. D. McDaniel. On the (statistical) detection of adversarial examples. *CoRR*, abs/1702.06280, 2017.

[13] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. D. Tygar. Adversarial machine learning. In *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*, AISec '11, pages 43–58, New York, NY, USA, 2011. ACM.

[14] S. Jia, L. Wang, and X. Li. View-invariant gait authentication based on silhouette contours analysis and view estimation. *IEEE/CAA Journal of Automatica Sinica*, 2(2):226–232, April 2015.

[15] A. Kantchelian, J. D. Tygar, and A. D. Joseph. Evasion and hardening of tree ensemble classifiers. *CoRR*, abs/1509.07892, 2015.

[16] L. Middleton, A. A. Buss, A. Bazin, and M. S. Nixon. A floor sensor system for gait recognition. In *Proceedings of the Fourth IEEE Workshop on Automatic Identification Advanced Technologies*, AUTOID '05, pages 171–176, Washington, DC, USA, 2005. IEEE Computer Society.

[17] H. Minh Thang, V. Quang Viet, T. Nguyen, and D. Choi. Gait identification using accelerometer on mobile phone, 11 2012.

[18] M. P. Murray. Gait as a total pattern of movement: including a bibliography on gait. *American Journal of Physical Medicine & Rehabilitation*, 1967.

[19] J. Newsome, B. Karp, and D. Song. Paragraph: Thwarting signature learning by training maliciously. In D. Zamboni and C. Kruegel, editors, *Recent Advances in Intrusion Detection*, pages 81–105, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[20] N. Papernot, P. D. McDaniel, and I. J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *CoRR*, abs/1605.07277, 2016.

[21] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013.

[22] T. Van hamme, D. Preuveneers, and W. Joosen. *Improving Resilience of Behaviometric Based Continuous Authentication with Multiple Accelerometers*, pages 473–485. Springer International Publishing, Cham, 2017.

[23] Y. Wang, J. Wong, and A. Miner. Anomaly intrusion detection using one class svm. In *Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop, 2004.*, pages 358–364, June 2004.

[24] Y. Q. Weilin Xu and D. Evans. Automatically evading classifiers: A case study on pdf malware classifiers. In *In Network and Distributed System Security Symposium 2016 (NDSS), San Diego, February 2016*, 2016.