Adversarial Transferability in Embedded Sensor Systems: An Activity Recognition Perspective

RAMESH KUMAR SAH, Washington State University, Pullman, United States
HASSAN GHASEMZADEH, Arizona State University, Tempe, United States

Machine learning algorithms are increasingly used for inference and decision-making in embedded systems. Data from sensors are used to train machine learning models for various smart functions of embedded and cyber-physical systems ranging from applications in healthcare, autonomous vehicles, and national security. However, recent studies have shown that machine learning models can be fooled by adding adversarial noise to their inputs. The perturbed inputs are called adversarial examples. Furthermore, adversarial examples designed to fool one machine learning system are also often effective against another system. This property of adversarial examples is called *adversarial transferability* and has not been explored in wearable systems to date. In this work, we take the first stride in studying adversarial transferability in wearable sensor systems from four viewpoints: (1) transferability between machine learning models; (2) transferability across users/subjects of the embedded system; (3) transferability across sensor body locations; and (4) transferability across datasets used for model training. We present a set of carefully designed experiments to investigate these transferability scenarios. We also propose a threat model describing the interactions of an adversary with the source and target sensor systems in different transferability settings. In most cases, we found high untargeted transferability, whereas targeted transferability success scores varied from 0% to 80%. The transferability of adversarial examples depends on many factors such as the inclusion of data from all subjects, sensor body position, number of samples in the dataset, type of learning algorithm, and the distribution of source and target system dataset. The transferability of adversarial examples decreased sharply when the data distribution of the source and target system became more distinct. We also provide guidelines and suggestions for the community for designing robust sensor systems. Code and dataset used in our analysis is publicly available here.[1]

CCS Concepts: • **Computer systems organization** → **Embedded systems** • **Computing methodologies** → *Machine learning algorithms*; *Neural networkss*;

Additional Key Words and Phrases: Sensor systems, human activity recognition, adversarial machine learning, adversarial transferability

ACM Reference Format:
Ramesh Kumar Sah and Hassan Ghasemzadeh. 2024. Adversarial Transferability in Embedded Sensor Systems: An Activity Recognition Perspective. *ACM Trans. Embedd. Comput. Syst.* 23, 2, Article 20 (March 2024), 31 pages. https://doi.org/10.1145/3641861

[1]https://github.com/rameshKrSah/adversarial-transferability-sensor-systems

This work was supported in part by National Science Foundation under grant CNS-1750679. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding organizations.

Authors' addresses: R. K. Sah, Washington State University, PO Box 641035 Pullman, WA, 99164, United States; e-mail: ramesh.sah@wsu.edu; H. Ghasemzadeh, Arizona State University, 1151 S Forest Ave Tempe, AZ, 85281, United States; e-mail: hassan.ghasemzadeh@wsu.edu.

ACM 1539-9087/2024/03-ART20
https://doi.org/10.1145/3641861

1 INTRODUCTION

Machine learning (**ML**) algorithms have increasingly become an integral part of embedded and cyber-physical systems. In the context of embedded systems, ML models are used to drive the inference and smart decision-making capabilities of embedded and wearable sensor systems. For example, data from sensors such as accelerometer and gyroscope are used for human activity recognition [34], images from camera are used to detect stop signs in self-driving cars [37], and sound signals are used for person detection in smart home applications [20]. Advances in sensor and computation technologies now allows for real-time continuous use of ML algorithms for smart functionalities inherent in modern embedded systems.

However, recent studies have found that an adversary can easily fool ML models with the addition of carefully computed perturbations to their inputs [7, 14, 33, 35]. These perturbed inputs are referred to as *adversarial examples*. Even the addition of a small but carefully computed perturbation to benign inputs, as shown in Figure 1, can degrade the performance of ML systems significantly [3, 21, 29, 33, 35]. What distinguishes adversarial perturbations from random noise is that adversarial examples are misclassified far more often than samples that have been perturbed by random noise, even if the magnitude of random noise is much larger compared to the adversarial perturbation [35]. The problem is further exacerbated by the fact that adversarial examples are highly transferable and adversarial examples computed to attack one ML system are often successful in fooling another ML system. The issue of adversarial examples raise serious concern about the security and reliability of ML algorithms and in-turn on embedded systems because of their reliance on ML algorithms for important functions. Studying adversarial examples and their properties in the context of embedded system is an important topic to safeguard and invent measures to make these systems secure and robust against adversarial attacks.

1.1 Motivation

First we establish the threat model used in this work and answer why the transferability of adversarial examples is crucial to the discussion of robustness in ML powered embedded sensor systems. Assume Bob, an adversary, has complete access to source system S_1 with ML model M_1 trained using dataset D_1. Alice is a system administrator who wants to protect the target system S_2 with ML model M_2 trained on dataset D_2 as shown in Figure 2. The dataset D_1 available freely can be accessed by anyone and dataset D_2 can be the same as D_1 or some private dataset only available to Alice. Also, models M_1 and M_2 can be of same or different types or architectures and have same or different hyper-parameter values. Bob has access to Alice's system via an oracle, and hence can submit inputs and observe outputs. Bob being an adversary wants to attack Alice's system, such that M_2 is fooled in classifying inputs into wrong classes. Bob can attack Alice's system in one of two ways. Bob can either compute adversarial examples using M_1 and D_1 and transfer them to M_2 in the hope of fooling M_2 or train a substitute model M_S on dataset D_S generated using the oracle and then use the substitute model to compute adversarial examples to fool M_2. In both cases, Bob tries to exploit the transferability property of adversarial examples to attack target system S_2. In this work, we explore different types of adversarial transferability inherent to embedded sensor systems Bob can exploit to attack Alice's system. In the discussion, that follows we recognize Bob's system as *Source System* and Alice's system as *Target System*. For all four adversarial transferability modes we have discussed in this work, Bob has complete access to source system S_1, but can only query target system S_2 on inputs and observe outputs.

Adversarial transferability captures the ability of an adversarial attack against an ML system to be effective against other independently trained systems [11]. The transferability of adversarial examples was first examined in [35], in which the authors studied adversarial transferability (1) between different ML models trained over the same dataset, and (2) between same or

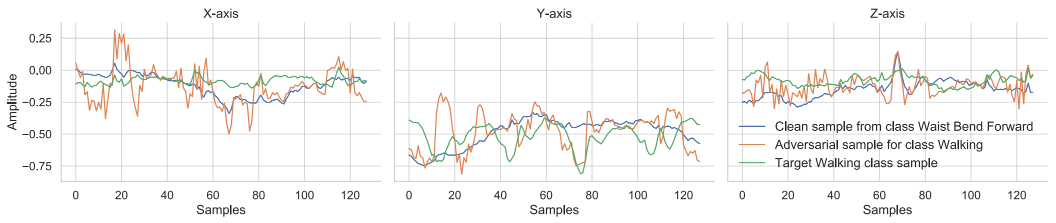


Fig. 1. 3-axial graphs for a benign sample take from the dataset, a targeted adversarial sample computed using the BIM, and a benign sample for the target class.

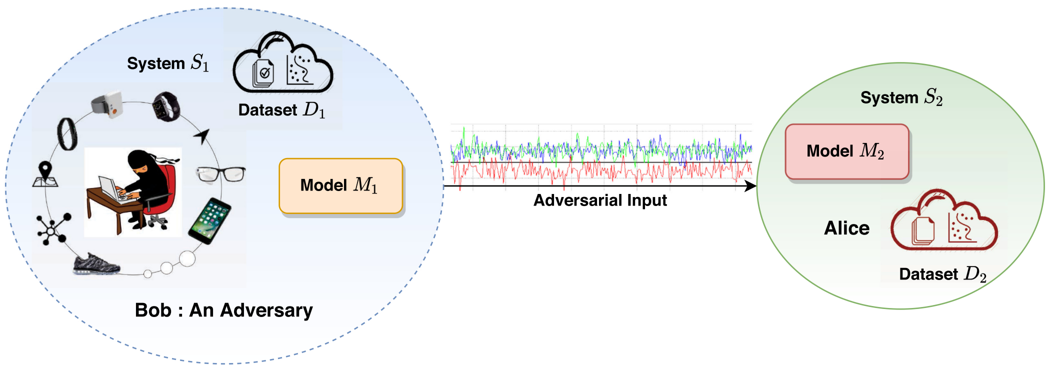


Fig. 2. The operating scenario for transferability of adversarial examples. Bob is an adversary with complete access to source system (S_1) and wants to attack Alice's system (S_2) by computing adversarial examples using the source system and transferring them to Alice's system.

different machine learning models trained over disjoint subsets of a dataset. Motivated by the results of [11], numerous studies have explored adversarial transferability for both test-time evasion attacks and training-time poisoning attacks [11, 25, 29, 33]. Furthermore, prior research has shown that adversarial examples can be generated in wearable sensor systems for human activity recognition [33]. However, the transferability of adversarial examples that take into account characteristics of embedded systems has not been studied yet, leaving a gap in the research, which we believe has significant and novel consequences. Because in addition to the traditional notion of transferability—between different models trained on the same or disjoint subsets of a dataset—we also need to consider new dimensions when exploring adversarial transferability in wearable sensor systems.

Embedded systems are dynamic in nature with properties impacting their operation. For example, consider the placement of a wearable device on the human body to detect human daily living activities. There are many devices available in the market today that can be worn on the body in various ways. Some are worn as a watch, others can be clipped onto clothes or shoes, strapped around the chest, and so forth. Furthermore, depending on the body location of the device, the sensor readings are very different, and consequently, ML algorithms trained on these sensor data learn unique mappings between inputs and outputs. Therefore an adversary who is planning to attack these types of ML systems must also take into account the different properties associated with them. These properties are well discussed in the literature [26, 41] for the case of building inference models. But to the best of our knowledge, there has not been any work that had discussed these properties of embedded systems from an adversarial point of view. All these lead to the fact that adversarial transferability in sensor systems is not simple and straightforward and has many nuances. We believe an extensive study of the adversarial transferability will not only show the strength of adversarial attacks but also mark their shortcomings and help us understand this unexplored problem space.

1.2 Contributions

In this article, we present the first comprehensive evaluation of the transferability of adversarial examples in the context of embedded sensor systems with wearable-based activity recognition as our pilot application. We not only consider the traditional notion of transferability but extend that with novel transferability directions unique to sensor systems. In particular, we discuss the transferability of adversarial examples from the following four perspectives:

—Adversarial transferability between ML models.
—Adversarial transferability across users.
—Adversarial transferability across sensor body locations.
—Adversarial transferability between datasets.

To this end, we make the following contributions to this work. We for the first time introduce and define novel types of adversarial transferability in the context of embedded sensor systems with a particular focus on wearable computing systems. Second, we conduct an extensive set of experiments that highlight vulnerabilities and strengths of embedded systems under different transferability cases for both targeted and untargeted evasion attacks. Third, we discuss and validate our results with theoretical and graphical interpretations that take into account the properties of both models and data distribution. Finally, we discuss open problems and possible research directions for adversarial transferability in general for sensor systems.

2 BACKGROUND

2.1 Human Activity Recognition Pipeline

The problem of human activity recognition can be defined as: Given a set $W = \{W_0, \ldots, W_{m-1}\}$ of m equally sized temporal window of sensor readings, such that each window W_i contains a set of sensor reading $S = \{S_{i,0}, \ldots, S_{i,k-1}\}$, and a set $A = \{a_0, \ldots a_n - 1\}$ of n activity labels, the goal is to find a mapping function $f : S_i \rightarrow A$ that can be evaluated for all possible values of S_i [22]. Raw data from sensors, such as accelerometer, gyroscope, and magnetometer, are collected and passed into the processing stage for filtering and noise removal. The next stage is segmentation, where a continuous stream of the sensor values is divided into temporal windows. After segmentation, statistical and structural features are extracted from each window segment and are used to train ML algorithms for activity classification. Another very successful approach to human activity classification uses **Convolutional Neural Network (CNN)** with raw sensor segments as inputs. CNN model learns the features and the classifier simultaneously during the training process from raw sensor data.

2.2 Adversarial Machine Learning

Given an ML classifier $f_\theta(x)$ characterized by the parameters θ and trained on dataset $D = \{(x, y)\}$, an adversary tries to find inputs that are formed by applying small but intentional perturbations (δ) to the original samples x such that the perturbed inputs $\bar{x} = x + \delta$ are almost indistinguishable from the original samples and result in the classifier predicting an incorrect label $\bar{y}$ with high confidence. These perturbed input samples are called *adversarial examples*. The objective of adversarial learning is to find perturbation δ which when added to the original inputs x changes the output of the classifier $f_\theta(\bar{x}) \neq y$. In general, an adversary can attack an ML system in three ways.

(1) *Poisoning Attacks*: In poisoning attacks, an adversary attempts to degrade the performance of an ML classifier by injecting adversarial examples during the training process to force the classifier to learn false connections between input and outputs.
(2) *Evasion Attacks*: Evasion attack is the most common type of adversarial attack carried out during test time and involves getting the target model to make mistakes on input samples. Evasion attack is sub-divided into two types: (1) untargeted attack, and (2) targeted attack. In an untargeted attack, an adversary intends to misclassify $\bar{x}$ into any class other than its true class i.e., $f_\theta(\bar{x}) \neq y$ such that $f_\theta(x) = y$. For a targeted adversarial example $\bar{x}$, the adversary defines the target class $\bar{y}$ in which it wants to have the target model classify the adversarial example.
(3) *Exploratory Attacks*: In exploratory attacks, the adversary tries to gain as much knowledge as possible about the learning algorithm of the target system and patterns in the training data.

We only consider evasion attack methods in our analysis, because we evaluate adversarial transferability at inference time. The difficulty in mounting evasion attacks against a target system is heavily influenced by the knowledge an adversary has about the target system. The extent of an adversary's knowledge about the target system dictates the setting in which it operates.

(1) *White-box Setting*: A white-box setting assumes that the adversary has complete knowledge about the target system. It includes anything related to the target system such as dataset, model architectures, hyper-parameters values, activation functions, number of layers, and model weight. This comprehensive knowledge about the target system makes it easier for the adversary to mount successful evasion attacks. In this mode, the adversary can compute adversarial examples using the target system and does not have to rely on the transferability property of adversarial examples.
(2) *Black-box Setting*: In the black-box setting, the adversary has no knowledge of the target system. The adversary only has access to an oracle to the target system to submit inputs and observe outputs. Evasion attacks in a black-box setting exploit the transferability properties of adversarial examples to mislead the target system.

The difficulty of operating in a black-box setting is mitigated by exploiting the transferability property of adversarial examples. In our threat model, adversary Bob operates in the white-box setting with respect to source system S_1 and black-box setting with respect to the target system S_2. Hence, Bob depends on the transferability property of adversarial examples computed using S_1 to fool the target system S_2.

2.3 Methods of Generating Adversarial Examples

The fundamental condition when computing adversarial example is that the perturbation $\delta = \{\delta_1, \delta_2, \ldots, \delta_n\}$ added to the benign samples $x = \{x_1, x_2, \ldots, x_n\}$ to get adversarial samples $\bar{x} = \{\bar{x_1}, \bar{x_2}, \ldots, \bar{x_n}\}$ cannot be large. This requirement is satisfied by bounding the adversarial perturbation δ with some adversarial budget ϵ using l_p-norms, where $p \in \{0, 1, 2, \infty\}$. For a model f_θ, adversarial examples $\bar{x}$ are defined as the solution to the following optimization problem:

$$\bar{x} = x + \underset{\delta}{\operatorname{argmin}}\{\|\delta\| : f(x+\delta) \neq f(x)\}S. \tag{1}$$

Here, $\|.\|$ is a type of l_p-norm defined by the method used to compute adversarial examples. For two vectors x and $\bar{x}$, l_0 counts the number of elements in $\bar{x}$ that has changed its values compared to x, l_2 measure the Euclidean distance between the two vectors, and l_∞ denotes the maximum changes for all elements in the vector $\bar{x}$.

2.3.1 Fast Gradient Sign Method (FGSM). FGSM was proposed in [14] and is one of the simplest and computationally efficient method to compute adversarial examples. FGSM computes the adversarial perturbation by calculating the gradient of the loss function of the model with respect to the input. This method solves the following optimization problem to maximize the loss such that adversarial perturbations are bounded by ϵ subject to l_∞ norm.

$$\bar{x} = x + \epsilon * \text{sign}(\nabla_x J_\theta(x, y)). \tag{2}$$

Here, J_θ is the loss function, and ∇_x denotes the gradient of the loss function with respect to the input x, and y is the actual label [9]. For targeted examples, FGSM minimizes the loss function with respect to input x such that the modified input is classified into the target class $\bar{y}$ specified by the adversary.

$$\bar{x} = x - \epsilon * \text{sign}(\nabla_x J_\theta(x, \bar{y})). \tag{3}$$

Notice the change in sign and also the presence of the target class $\bar{y}$ in the optimization Equation (3). For targeted case, we are trying to find adversarial perturbations δ that decrease the loss of the model for the target class $\bar{y}$ and for untargeted case we find adversarial perturbations which increase the loss of the model in general.

2.3.2 Basic Iterative Method (BIM). BIM is an extension to the fast gradient sign method and runs FGSM n number of times with a small step size. Iteratively running FGSM allows the adversary to search the model input space thoroughly to find optimal perturbations.

$$\begin{aligned} \bar{x}_0 &= x, \\ \bar{x}_{n+1} &= \text{Clip}_{x,\epsilon}\{\bar{x}_n + \alpha * \text{sign}(\nabla_x J_\theta(x, y))\}. \end{aligned} \tag{4}$$

Here, α is the step size and is usually defined as $\alpha = \epsilon/n$. $\text{Clip}_{x,\epsilon}(A)$ denotes the element-wise clipping of A, such that the range of $A_{i,j}$ after clipping is in the interval $[x_{i,j} - \epsilon, x_{i,j} + \epsilon]$. The BIM can also be used to compute targeted adversarial examples by the simple modification of sign reversal and the introduction of the target class in Equation (4).

2.3.3 Jacobian-based Saliency Map Attack (SMM). Jacobian-based saliency map attack (SMM) [30] finds features of input x that cause the most significant changes to the output of the model. SMM computes perturbations that induce significant output variations such that a change in a small portion of features of x foold the target model [39]. SMM computes the Jacobian matrix of the given input x to determine adversarial perturbations.

$$J_F(x) = \frac{\partial F(x)}{\partial x} = \left[\frac{\partial F_j(x)}{\partial x_i}\right]_{ixj}. \tag{5}$$

Here, F is the second-to-last layer logits of the neural network.

2.3.4 Carlini-Wagner (CW) Attack. The CW attack solves the following optimization problem to find adversarial perturbations.

$$\begin{aligned} &\min \|\delta\|_p \\ \text{subject to} \quad & C(x + \delta) = t, \quad x + \delta \in [0, 1]^n \end{aligned} \tag{6}$$

where $C(x)$ is the class label returned for input x and the noise level is measured using either l_0, l_2 or l_∞ norm. CW attack is considered one of the best evasion attack method and computes adversarial examples by finding the smallest noise $\delta \in R^{nxn}$ that changes the classification of the model to a class t.

2.3.5 Momentum Iterative Attack (MIM). Momentum iterative attack method [12] integrates the concept of momentum into the BIM to generate adversarial examples for targeted and untargeted cases using l_2 and l_∞ norms respectively. The momentum is a technique for accelerating gradient descent algorithms by accumulating a velocity vector in the gradient direction of the loss function across iterations [12]. The introduction of momentum helps the method achieve optimum results faster by stabilizing update directions and escaping from poor local maxima.

3 APPROACH

We use human activity recognition as an example case for demonstration and validation of our experiments. Human activity classification involves some type of sensor system such as a smartwatch, smartphone, smart shoes, chest band, or fitness band to detect and measure physical activities. The underlying ML algorithms use the data from sensors to learn the characteristics of different activities. But depending on the properties of the sensors, the sensor reading can differ significantly even though the sensors are trying to measure and detect the same physical phenomena. This is because human activities are highly complex and dynamic processes dependent upon various factors. The sensor readings for an activity vary significantly even if the same person performs the same activity under similar conditions compared to say image classification where an image of a dog is always a dog independent of the presentation and context. These variations in the sensor reading result in the trained ML algorithms learning unique mappings between inputs and outputs, creating newer challenges and opportunities for an adversary that wants to attack these systems.

3.1 Adversarial Transferability

From differences in the electrical properties of the sensor to the location of sensor on the human body, there are numerous ways in which different aspects of the sensor systems can affect adversarial transferability. Therefore, in this work, we study adversarial transferability from the following four perspectives:

- **Adversarial transferability between ML models:** The transferability between different ML models trained on the whole or subset of the same dataset is the default and the most discussed variety of adversarial transferability. To exploit this mode of transferability, the adversary computes adversarial examples using one ML model and then performs adversarial attacks on other models using the generated adversarial examples.
- **Adversarial transferability across users:** In sensor systems, the dataset used to train ML algorithms is collected using human subjects. This is similar to an image dataset where images of various objects are captured using different types of cameras. But what separates human subjects from the optical sensor in cameras is that human subjects inject biases in the data that are personalized to each individual and are hard to eliminate with preprocessing. Attributes associated with individuals give the problem of adversarial transferability in sensor systems a new direction. The adversary can leverage the biases injected by individuals to design better attack methods or suffer from this when trying to attack a target system. Hence, evaluating adversarial transferability between ML systems trained on data from different individuals becomes crucial.
- **Adversarial transferability across sensor body locations:** Another important attribute of wearable sensor systems is the body location of the sensor. For example, activity trackers can be worn in many different ways. Some can be worn as a wristwatch or wristband, others can be clipped onto clothes and shoes or placed inside pockets, and some can be even worn as jewelry. For two sensors of the same type—one wrapped around the subject's chest and

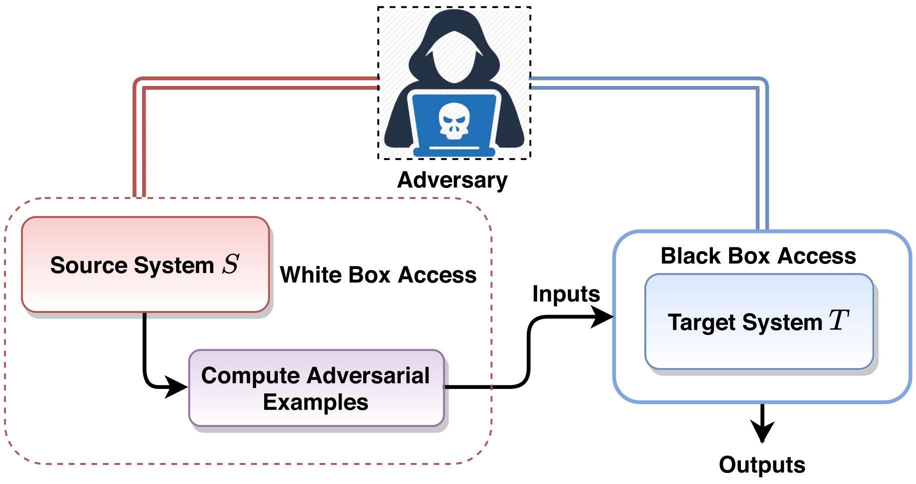


Fig. 3. Threat model in which an adversary operates with complete access to the source system S but only oracle access to the target system T. The adversary has no knowledge about the target system and its properties, such as dataset used for training, type of learning algorithm, and hyper-parameters of the model.

other worn on the wrist—the sensor readings depend heavily on the orientation of the sensor and placement. These differences in the sensor readings affect the mapping learned by ML model and consequently transferability of adversarial examples.

- **Adversarial transferability between datasets:** The final and most complex type of adversarial transferability is transferability between ML systems—same or of different architectures—trained on different datasets. For example, in human activity recognition different manufacturers use different types of sensors and collect proprietary datasets to train ML algorithms. Now for an adversary which has access to system from one company, it is challenging to attack a system from another manufacturer. The challenges can stem from subject bias, sensors position, types of sensors, and data processing routines.

3.2 Threat Model

Depending on the type of adversarial transferability the adversary wants to exploit to attack the target system, the adversary operates in different settings. In general, the threat model has two main components. The first part concerns the target system T, which the adversary wants to attack and the second part takes into account the source system S. The adversary can only send inputs to the target system and observe the class prediction, and hence operates in the black-box setting with respect to the target system T. The adversary has white-box access to the source system S and can compute adversarial examples using the source model S_m. Here the objective of the adversary is to fool the target system T on the adversarial examples computed using the source model S_m. Figure 3 shows the graphical representation of the threat model. In all four cases of adversarial transferability, the adversary computes adversarial examples using the source model S_m trained on source dataset D_S and attacks the target system T with target model S_t trained on dataset D_T. In transferability between ML models D_S and D_T are same and for remaining cases of adversarial transferability datasets D_S and D_T are different. In adversarial transferability across users, the source dataset contains samples from a group of subjects and the target dataset contains samples from the remaining subjects. In adversarial transferability across sensor body locations, the source dataset contains sensor reading from sensor placed at one body position, for example right-wrist,

Table 1. Characteristics of the Three Datasets Used in the Analysis (# Denotes "Number of")

Dataset	# Subject	# Activities	Frequency	Window Size (Seconds)	# Devices	# Samples
UCI	30	6	50 Hz	2.56	1	10,299
MHEALTH	10	12	50 Hz	2.56	3	5,133
DL	7	6	50 Hz	2.56	2	16,434

and the target dataset contains sensor values from sensor placed at another body position, say chest. Finally, in adversarial transferability between datasets, the source and target dataset are completely different and have different distributions.

3.3 Measuring Adversarial Transferability

Measuring adversarial transferability means we want to determine how many adversarial examples designed for the source system can fool the target system. For untargeted attacks, we want to quantify how many input samples were classified into any class other than the ground truth class and for targeted attack we want to measure how many samples were classified into the target class. We introduce a new metric called *Success Score* (*SC*) to measure adversarial transferability in both untargeted and targeted cases. Success score defined in percentage is the ratio of the number of adversarial examples that were able to fool the target system (N_t) to the total number of samples (N)

$$\text{Success Score (SC)} = \frac{N_t}{N} * 100. \tag{7}$$

For untargeted case, N_t is equal to the number of adversarial examples that are misclassified and for targeted case N_t equals the number of adversarial examples that are classified into the target class. Misclassification means the output label assigned to adversarial examples is different from label assigned to clean samples used to generate adversarial examples. In the targeted case, success score is computed for adversarial examples generated from clean samples which are not already classified into the target class. Furthermore, in both targeted and untargeted cases success score is only computed for samples that failed to fool the target model without addition of adversarial perturbation.

3.4 Datasets

In our experiments, we have used three real-world human activity recognition datasets Table 1. We have conducted our experiments with 3-axial accelerometer data.

— *UCI dataset*[2] [2] was compiled from a group of 30 participants, each wearing a smartphone on their waist and performing six different activities in a lab setting. Data from 3-axial accelerometer and gyroscope sensors were sampled at a frequency of 50 Hz and pre-processed to remove noise.

— *MHEALTH dataset*[3] [5] consists of body motion and vital signs recording of 10 volunteers of different profiles while performing 12 different physical activities in an out-of-lab environment without any constraint, with the exception that the subject should try their best when executing them. *Shimmer2* wearable device placed on the subject's chest, right wrist, and left ankle were used to measure the motion experienced by the diverse body parts using a 3-axial accelerometer at a frequency of 50 Hz. The class *Jump Front and Back* has

[2]https://archive.ics.uci.edu/ml/datasets/human+activity+recognition+using+smartphones
[3]https://archive.ics.uci.edu/ml/datasets/MHEALTH+Dataset

fewer number of samples compared to other classes. Therefore, to balance the dataset, we have removed the samples from the *Jump Front and Back* class from our analysis.

- **Daily Log** (**DL**) dataset[4] [36] has accelerometer, orientation, and GPS sensor data collected from 7 individuals using a smartphone and smartwatch with a self-developed sensor data collector and labeling framework. Acceleration and orientation sensors were sampled at 50 Hz and GPS data was collected every 10 minutes. The data was collected when participants were doing their daily routine and it was up the participants where the device should be positioned on the body. We randomly select subset of the data to use in our experiments such that each activity class has the same number of samples.

Now we establish some conditions so that the analysis of different types adversarial transferability across the datasets is possible and sound.

(1) **Sampling Frequency:** One of the criteria we used to select datasets for our experiments is the sampling frequency of the sensor. For all real-world datasets used in this article, the sampling frequency is 50 Hz, which is considered adequate for human activity recognition [24].
(2) **Input Size:** The length of the input window segment in all datasets must be the same because we cannot train ML algorithms with variable input sizes. In our experiments, we have set the length of the raw sensor segment to 128, which corresponds to the window size of 2.56 seconds at a sampling frequency of 50 Hz. Setting the window size to 128 was motivated by the fact that a window size of 1–2 seconds with 50% overlap is considered a good choice for activity classification [4].
(3) **Data Scaling:** The range of values in the three datasets are very different. We used the *MinMaxScaler* with range set to [−1.0, 1.0] from the *sklearn* library [31] to standardize all three datasets. *MinMaxScaler* is the least disruptive to the information in the original data and preserves the shape of the data and does not reduce the importance of outliers.
(4) **Activity Classes:** Another important factor when choosing datasets for our experiments was the activity classes. The baseline condition was that there should be some activity classes that are common for all datasets allowing us to analyze targeted adversarial transferability between datasets. The activities *walking, sitting, standing,* and *climbing stairs (walking up)* are common to all three datasets. Also, having activities classes that are not common between the datasets further helps us analyze transferability with generalization.

4 EXPERIMENTAL RESULTS

In this section, we discuss our experiments and results. We discuss four cases of adversarial transferability, and for each case, we present results for both targeted and untargeted evasion attacks. The *CleverHans* [27] library was used to compute adversarial examples with the following parameters: (1) adversarial perturbation budget from the set $\epsilon \in [0.1, 0.25, 0.5, 0.9]$, (2) range clipping of adversarial examples set to $[-1.0, 1.0]$, (3) number of iterations for BIM and momentum iterative method set to 50. Also, for iterative methods, the perturbation budget per iteration is $\epsilon/50$.

4.1 Adversarial Transferability Between Machine Learning Models

To analyze the adversarial transferability between ML models, we trained six different ML algorithms for a common dataset and computed adversarial examples using one of the trained model. We used the feature data of all three datasets for training the ML algorithms. Using the feature

[4]https://sensor.informatik.uni-mannheim.de/#dataset_dailylog

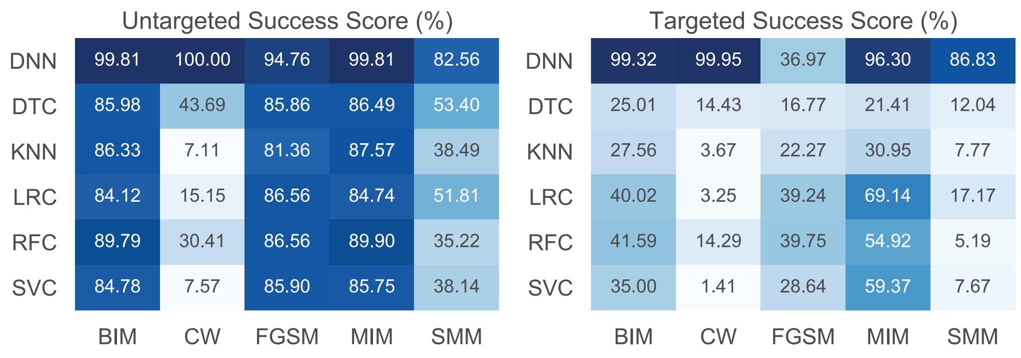
Table 2. Classification Accuracy of Different ML Algorithms on the Training and Test Set of all Three Datasets

ML	UCI		MHEALTH		DL	
Algorithms	Train Set	Test Set	Train Set	Test Set	Train Set	Test Set
SVC	76.20%	76.38%	90.40%	90.46%	87.79%	87.61%
RFC	100.0%	84.85%	100.0%	96.39%	100.0%	89.87%
KNN	84.85%	79.10%	97.56%	96.07%	91.96%	87.90%
DTC	100.0%	72.93%	100.0%	92.22%	100.0%	85.08%
LRC	75.49%	76.54%	91.15%	89.90%	86.12%	85.66%
DNN	84.90%	82.05%	99.25%	97.19%	94.75%	89.82%

data enabled us to train different kinds of ML algorithms, which is not possible using the raw sensor data. We computed 45 statistical features commonly used in HAR [40], from sensor segments of all three datasets. We selected the following algorithms to evaluate adversarial transferability between ML models: (1) **Support Vector Classifier** (**SVC**), (2) **Random Forest Classifier** (**RFC**), (3) **K-Nearest Neighbor** (**KNN**) Classifier, (4) **Decision Tree Classifier** (**DTC**), (5) **Logistic Regression Classifier** (**LRC**), and (6) **Deep Neural Network** (**DNN**). The DNN has three layers with 64, and 32 neurons in the first and second layers. In the last layer the number of neurons is equal to the number of activity classes for the respective dataset. $l2$-regularization with coefficient 0.001 and ReLU activation is used in the first and second layers, and the output layer has Softmax activation. *TensorFlow* [1] was used to train the DNN with hyper-parameters: 200 epoch, mini-batch size of 32, Adam [18] optimizer with learning rate 0.001, and sparse categorical cross-entropy loss. All other classifiers were trained using the sklearn library [31]. The maximum iteration for SVC was set to 5,000 with scaled gamma, and the number of estimators for RFC was set to 100. For logistic regression, the LBFGS solver was used with 5,000 maximum iterations and for KNNs the number of components was set to 5. All other parameters of classifiers were left to their default values.

Table 2 shows the classification accuracy of all trained models on the training and test set for the three datasets. In general, all trained models have very high classification accuracy on training and test sets. To evaluate these classifiers for adversarial transferability between ML models, we computed targeted and untargeted adversarial examples using the DNN model—the source model. We choose the DNN model, because evasion attacks methods based on gradient optimization are more mature and there are large number of successful attack methods available for neural networks [8, 14]. There are some adversarial attack methods that can compute adversarial examples with non-parametric models such as decision trees and KNNs [29, 38] and we explore these attacks later in our discussion section. For targeted attacks, we selected the "*Sitting*" activity class as the target class because it is common across all three datasets.

Figures 4 and 5 show the success score of untargeted and targeted adversarial examples for the UCI and DL datasets. Different adversarial attack methods were used to generate adversarial examples for the perturbation budget of $\epsilon = 0.5$ using the DNN model. Each number in the heatmap, shows the success score of adversarial examples computed using the attack method specified by the column index on the ML model denoted by the row index. For example, in Figure 4 the success score of untargeted adversarial examples computed with adversarial attack BIM on the SVC model is 84.78% and the success score of targeted adversarial examples is 35%. The results for the MHEALTH dataset was found to be very similar to the UCI dataset, and is presented in supplementary section.



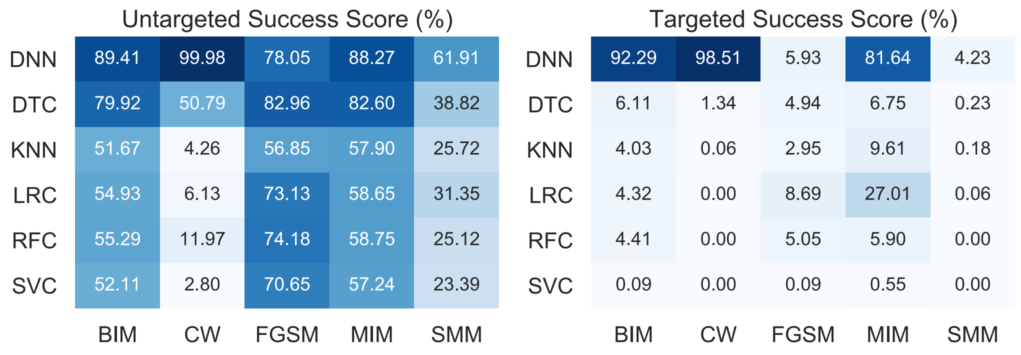
Untargeted Success Score (%)

	BIM	CW	FGSM	MIM	SMM
DNN	99.81	100.00	94.76	99.81	82.56
DTC	85.98	43.69	85.86	86.49	53.40
KNN	86.33	7.11	81.36	87.57	38.49
LRC	84.12	15.15	86.56	84.74	51.81
RFC	89.79	30.41	86.56	89.90	35.22
SVC	84.78	7.57	85.90	85.75	38.14

Targeted Success Score (%)

	BIM	CW	FGSM	MIM	SMM
DNN	99.32	99.95	36.97	96.30	86.83
DTC	25.01	14.43	16.77	21.41	12.04
KNN	27.56	3.67	22.27	30.95	7.77
LRC	40.02	3.25	39.24	69.14	17.17
RFC	41.59	14.29	39.75	54.92	5.19
SVC	35.00	1.41	28.64	59.37	7.67

Fig. 4. Success score of adversarial examples computed with the UCI dataset for transferability between models.



Untargeted Success Score (%)

	BIM	CW	FGSM	MIM	SMM
DNN	89.41	99.98	78.05	88.27	61.91
DTC	79.92	50.79	82.96	82.60	38.82
KNN	51.67	4.26	56.85	57.90	25.72
LRC	54.93	6.13	73.13	58.65	31.35
RFC	55.29	11.97	74.18	58.75	25.12
SVC	52.11	2.80	70.65	57.24	23.39

Targeted Success Score (%)

	BIM	CW	FGSM	MIM	SMM
DNN	92.29	98.51	5.93	81.64	4.23
DTC	6.11	1.34	4.94	6.75	0.23
KNN	4.03	0.06	2.95	9.61	0.18
LRC	4.32	0.00	8.69	27.01	0.06
RFC	4.41	0.00	5.05	5.90	0.00
SVC	0.09	0.00	0.09	0.55	0.00

Fig. 5. Success score of adversarial examples computed with the DL dataset for transferability between models.

In general, we found high adversarial transferability between ML models for untargeted adversarial examples. For targeted attacks, adversarial examples were less transferable for all three datasets. In particular, we found DTC, KNN, and RFC classifiers to be more robust towards targeted adversarial examples for UCI and MHEALTH datasets. We also found that the level of adversarial transferability between ML systems differed greatly across the three datasets. For the DL dataset, both targeted and untargeted adversarial examples were less likely to be transferable with targeted transferability success score values of 0.0% in many cases. We believe the lower success score of targeted adversarial examples in general is due to fundamental differences between the targeted and untargeted attacks. An untargeted attacks is considered successful if an input is classified into any class other than its actual class but for the targeted attack to be successful the input must be classified into the target class by the target system. Hence, targeted attack are much more difficult and an adversary will have higher success score for untargeted attack compared to targeted attack at the same level of perturbation budget and source and target models attributes. Also, we suspect the lower success score for the DL dataset is due to nature of the dataset. The DL dataset was collected in daily-living conditions while participants were following their daily routine with greater degree if flexibility compared to the MHEALTH and UCI dataset. Collecting sensor data in daily-living conditions can induces noise and artifacts in the sensor data and as a result different learning algorithms will learn different mappings between input and output. Furthermore, no data preprocessing is applied to the DL dataset but both UCI and MHEALTH dataset undergo noise removal and filtering. Consequently, adversarial transferability which aims to capitalize on

Table 3. Details about the Source and Target Dataset Fashioned by Randomly Selecting Subjects Data for the UCI, MHEALTH, and DL Datasets

Dataset	# Samples	# Subjects	# Source Subjects	# Source Samples	# Target Subjects	# Target Samples
UCI	10,299	30	15	5,138	15	5,161
MHEALTH	5,133	10	5	2,464	5	2,527
DL	16,434	7	3	9,918	4	6,516

Here, # means "number of".

Table 4. The Classification Accuracy of Source and Target Models on the Training and Test Set of all Three Datasets

Machine Learning	UCI		MHEALTH		DL	
System	Source Set	Target Set	Source Set	Target Set	Source Set	Target Set
Source	81.49%	61.42%	99.66%	66.55%	81.74%	25.28%
Target	62.43%	85.58%	81.37%	99.72%	34.09%	86.05%

the common input-output mappings shared by different ML algorithms to fool a target system on adversarial examples computed using the source system suffers greatly.

4.2 Adversarial Transferability Across Users

All three datasets have subject ID associated with each row of sensor readings or data files. We randomly selected data from half the subjects to create the dataset for the source system (source dataset) and the data from the remaining half subjects is used in the target system (target dataset). Table 3 shows the properties of source and target sets for all three datasets. We also decided to use 1−D CNN for both source and target system ML algorithm because of its simplicity and superior performance. CNN allows us to use the raw sensor data directly to train the model without needing to compute features from the sensor segments. The input CNN layer has 100 filters, kernel size of 10, and strides of 2. The second CNN layer layer has 50 filters and kernel size of 5. The third layer is a 1−D Global Max Pooling, which is followed by a fully-connected layers with 64 neurons and drop-out coefficient of 0.3. The last layer is also a fully-connected layer with the number of neurons equal to the number of activity class defined by the dataset. ReLU activation is used in all layers except the output layer, which uses Softmax activation function. The CNN model is trained using the Adam [18] optimizer with a learning rate of 0.001. The loss of the model is computed using the categorical cross-entropy loss function.

Table 4 shows the classification performance of the source and target system of UCI, MHEALTH, and DL datasets on their respective source and target sets. For UCI and MHEALTH datasets, the source and target models have high classification accuracy on both source and target datasets. High classification accuracy between source and target systems demonstrates high level of generalization between source and target systems. Therefore, in theory cross user adversarial transferability should be high for the UCI and MHEALTH datasets because source and target systems share common knowledge and an adversary should be able to exploit these common mappings to fool the target system using adversarial examples computed using source system. On the other hand, the classification accuracy is low for the DL dataset implying less shared knowledge between source and target systems and consequently predicting poor adversarial transferability.

Figures 6 and 7 show the success score of untargeted and targeted adversarial examples for UCI and DL datasets computed using five attack methods at four values of adversarial perturbation budgets $\epsilon \in \{0.1, 0.25, 0.5, 0.9\}$. For targeted attacks, the activity class "*Sitting*" is used as the target class. Untargeted adversarial examples were highly transferable to the target system in all

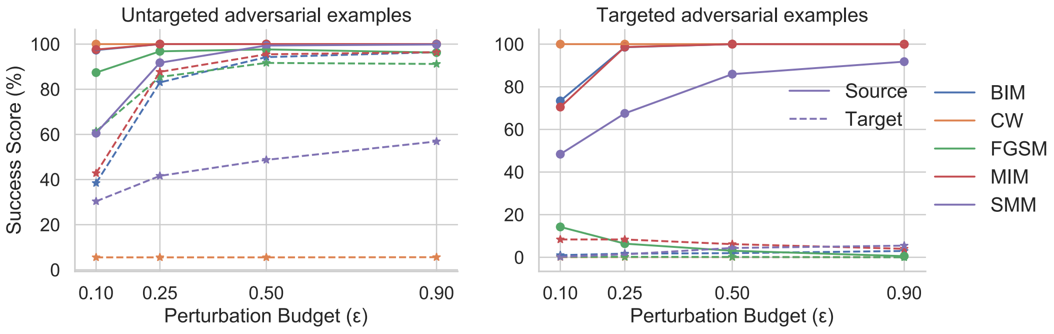


Fig. 6. Success score of adversarial examples on source and target systems with the UCI dataset for transferability across users.

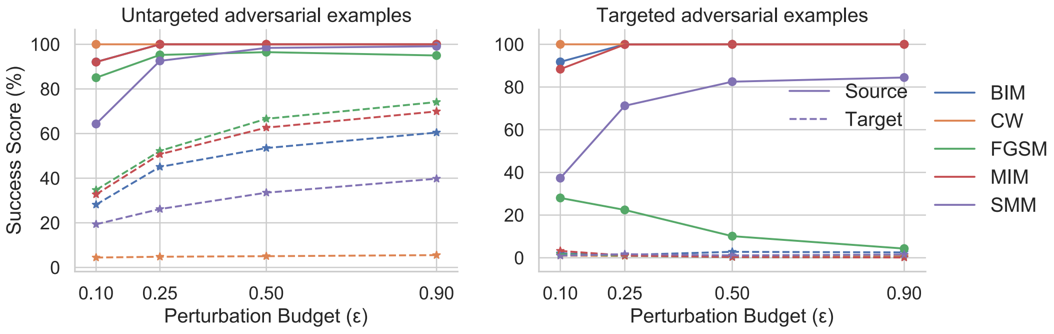


Fig. 7. Success score of adversarial examples on source and target systems with the DL dataset for transferability across users.

three cases. But untargeted success score for the DL dataset was lower compared to the UCI and MHEALTH dataset, indicating consequent of low generalization we observed between the source and target system for the DL dataset. Targeted adversarial examples were unsuccessful in all three cases, confirming that the individual characteristics encoded in the sensor data from each subject can greatly affect the adversarial transferability. The results for MHEALTH dataset is available in the supplementary materials.

4.3 Adversarial Transferability Across Sensor Body Locations

The MHEALTH dataset has readings from accelerometers placed at three different body positions. The first sensor is wrapped around the subject chest, the second is worn by the subject on the right wrist, and the last one is worn on the left ankle. All sensors have same physical and electrical characteristics. To evaluate adversarial transferability across sensor body locations, we perform experiments with different choice of sensor locations for the source and target systems. In the first case, the data from the chest sensor is used to train the source system and the data from the ankle sensor is used to train the target system. In the second case, the data from the wrist sensor is used to train the source system and the data from the chest sensor is used in the target system. Finally, we have data from ankle sensor used in the source system and data from wrist sensor is used to train the target system. Also, we use the same architecture of CNN used for transferability across subjects for the source and target models.

Table 5. The Classification Accuracy of Source and Target Systems on the Source and Target Datasets for all Three Cases of Sensor Body Positions

Machine Learning	Chest Vs. Left-Ankle		Right-Wrist Vs. Chest		Left-Ankle Vs. Wrist	
System	Source Set	Target Set	Source Set	Target Set	Source Set	Target Set
Source	98.81%	12.20%	99.67%	18.69%	98.55%	22.06%
Target	18.75%	96.75%	23.58%	99.17%	19.89%	99.43%

For example, the table for **Chest - Ankle** shows the classification accuracy of the Chest source system and Left-Ankle target system on both chest and left-ankle datasets.

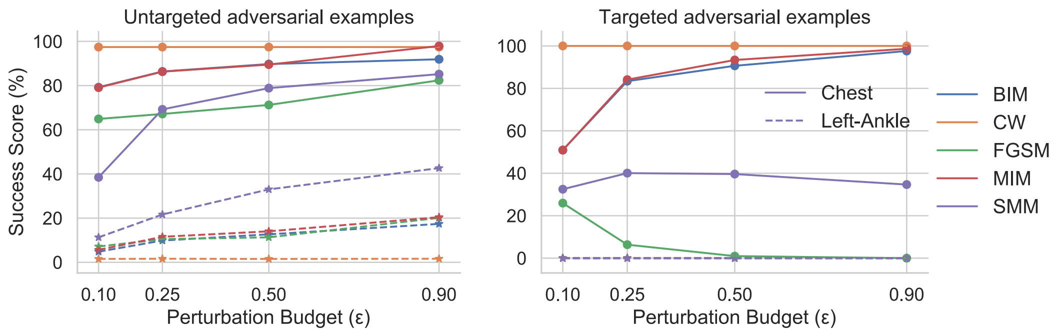


Fig. 8. Success score of adversarial examples computed using the source system (**Chest**) on source and target (**Left-Ankle**) systems.

After training the source and target model on their respective datasets obtained from sensors placed at different body locations, we evaluate the trained source and target models on both datasets. Table 5 shows the classification accuracy of these models on both datasets. In all cases, the classification accuracy of the source model on the target dataset and the target model on the source dataset is low, indicating low generalization between the source and target systems.

4.3.1 Chest Vs. Left-Ankle. Figure 8 shows the success score of untargeted and targeted adversarial examples computed using the chest (source) system on the chest and left-ankle (target) system. We found good transferability for untargeted attacks and very low transferability for targeted attacks for the target activity class of "Sitting". Untargeted adversarial examples with success score upto 100% on the source model performed fairly well, success score in the range 0%–40%, on the target model. The adversarial transferability further decreased for targeted attacks with success score of almost 0% on the target system while the success score was in the range of 20%–100% on the source system.

4.3.2 Right-Wrist Vs. Chest. Figure 9 shows the success score of untargeted and targeted adversarial examples computed using the right-wrist (source) system on the right-wrist and chest (target) system. We found high adversarial transferability for both untargeted and targeted attacks, with untargeted success score upto >90% and targeted success score upto 80% for the target class of "Sitting" on the target system.

The above results show that adversarial transferability differs greatly with the sensor body locations for source and target systems. If the sensors for the source and target system are located near each other, for example, in the case of *Right-Wrist Vs. Chest*, adversarial examples were able to fool the target system fairly well. But for source and target sensors placed far-apart on the body, in the case of *Chest Vs. Left-Ankle* and *Left-Ankle Vs. Right-Wrist*, the transferability of adversarial examples was low. Specifically, the success score of targeted adversarial examples was almost 0%

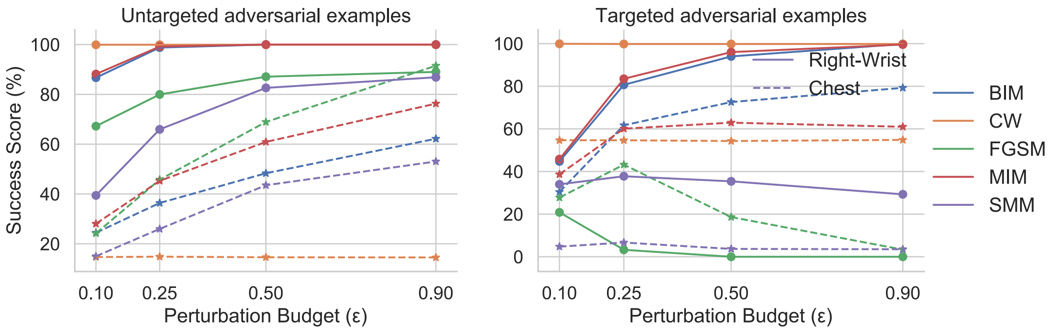


Fig. 9. Success score of examples computed using the source system (**Right-Wrist**) on source and target (**Chest**) systems.

Table 6. Success Score of Adversarial Examples Computed Using the Source (UCI) System on the Source and Target (MHEALTH) Systems

Evasion Attack Methods	Untargeted Attack Perturbation Budget (ϵ)							
	0.1		0.25		0.5		0.9	
	Source	Target	Source	Target	Source	Target	Source	Target
FGSM	74.09	0.11	84.15	0.85	87.65	11.96	89.74	36.15
BIM	86.91	0.19	96.89	0.62	97.86	8.03	97.94	25.78
MIM	86.79	0.19	96.07	0.97	96.50	13.09	96.07	40.23
SMM	51.84	0.11	78.17	0.11	92.73	0.11	96.07	0.11
CW	100.0	0.03	100.0	0.03	100.0	0.03	100.0	0.03
Evasion Attack Methods	Targeted Attack Perturbation Budget (ϵ)							
	0.1		0.25		0.5		0.9	
	Source	Target	Source	Target	Source	Target	Source	Target
FGSM	8.62	0.0	3.82	0.0	1.02	0.0	0.55	0.0
BIM	64.83	0.0	92.39	0.0	99.02	0.0	99.95	0.0
MIM	58.58	0.0	91.18	0.0	99.95	0.0	100.0	0.0
SMM	45.10	0.0	49.02	0.0	52.0	0.0	35.68	0.0
CW	99.95	0.0	99.95	0.0	99.95	0.0	99.95	0.0

for all attack methods at all values of adversarial perturbation budgets. The results for the case *Left-Ankle Vs. Right-Wrist* is provided in supplementary section.

4.4 Adversarial Transferability Between Datasets

Transferability between datasets includes all other types of transferability we have discussed so far and augments that with new variables such as sensor types, electrical properties of the sensor, and data processing steps. To evaluate adversarial transferability between datasets, we train source and target CNN models of same architecture and hyperparameters on different datasets. Since, we have three different datasets to evaluate adversarial transferability we have three different combinations for evaluation. Each of the combination assigns different dataset to the target and source systems. In the first experiment, we assigned the UCI dataset to the source system and the MHEALTH dataset to the target system. Table 6 shows the success score of untargeted and targeted adversarial examples for this case. In the second case, the DL dataset was assigned to the source system and the target system was trained on the UCI dataset. Table 7 shows the success score of untargeted and targeted adversarial examples for the second case. Finally, in the third case the

Table 7. Success Score of Adversarial Examples Computed Using the Source (DL) System on the Source and Target (UCI) Systems

Evasion Attack Methods	Untargeted Attack Perturbation Budget (ϵ)							
	0.1		0.25		0.5		0.9	
	Source	Target	Source	Target	Source	Target	Source	Target
FGSM	69.53	11.80	91.92	27.88	94.42	51.20	90.84	82.52
BIM	91.92	7.08	99.48	13.75	99.48	33.43	99.48	55.09
MIM	93.01	8.51	99.48	20.32	99.48	41.32	99.48	74.73
SMM	51.47	25.28	83.96	38.69	93.59	41.98	95.27	48.50
CW	100.0	5.47	100.0	5.59	100.0	5.84	100.0	6.01
Evasion Attack Methods	Targeted Attack Perturbation Budget (ϵ)							
	0.1		0.25		0.5		0.9	
	Source	Target	Source	Target	Source	Target	Source	Target
FGSM	0.43	11.51	0.17	4.03	0.0	4.34	0.0	2.04
BIM	52.24	11.70	67.64	9.32	90.62	5.81	97.02	4.17
MIM	56.22	7.41	71.52	3.90	98.30	5.09	99.97	5.25
SMM	4.05	2.26	3.73	1.07	1.92	0.11	0.67	0.05
CW	99.85	1.43	99.85	1.46	99.82	1.41	99.82	1.46

source system was trained on the MHEALTH dataset and the target system was trained on the DL dataset. The result for the third case can be found in the supplementary section. The success score were very similar to results for the first case of UCI source dataset and MHEALTH target dataset. We found poor adversarial transferability in the first case with highest untargeted success score of 40.23% at highest perturbation budget. The targeted success scores was 0% for all configurations. In the second case, we found untargeted success score up to 82% and highest targeted success score of 11.70%. The low adversarial transferability observed in this case demonstrates that with greater distinction between source and target systems the adversarial transferability decreases sharply. One interesting thing to note here is that targeted adversarial examples were more transferable at lowest adversarial perturbation budget ($\epsilon = 0.1$) and untargeted adversarial examples were most transferable at highest value of adversarial perturbation budget ($\epsilon = 0.9$).

5 DISCUSSION

In this section, we discuss our results and provide theoretical and graphical explanations. We generate adversarial examples using non-parametric ML algorithms such as DTC and KNN classifier and measure adversarial transferability. We also discuss adversarial transferability through the lens of non-robust features and manifold learning to provide explanation for our results and establish ideas for future research. Discussion on manifold learning is presented in the supplementary section.

5.1 Adversarial Attacks with Decision Trees and K-Nearest Neighbors

In our analysis, we found KNN and DTC classifiers to be robust against targeted adversarial examples computed using a DNN compared to other learning algorithms such as SVC and LRC. To further evaluate the robustness KNN and DTC algorithms, we computed targeted and untargeted adversarial examples using the KNN and DTC at the adversarial perturbation budget of $\epsilon = 0.5$. We used the **Region-based Attack** (**RBA**) [38] and heuristic decision tree attack (Papernot) [28] to compute adversarial examples using the decision tree classifier. The RBA finds the closet polyhedron to an input where the classifier predicts a label other than the actual label and outputs

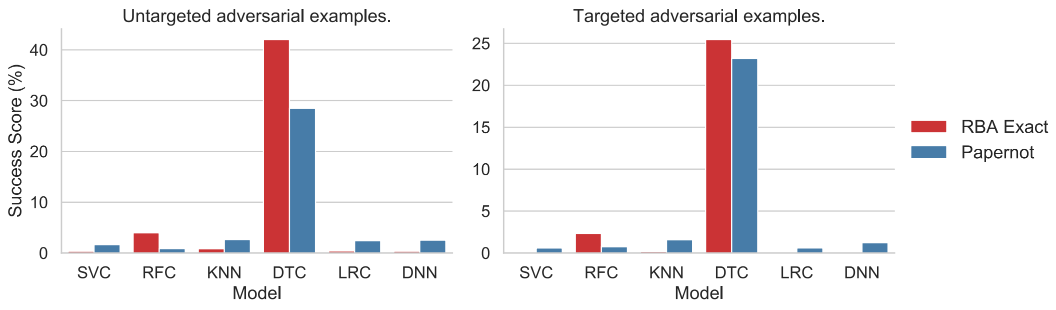


Fig. 10. Success score of adversarial examples computed using the DTC on different ML models.

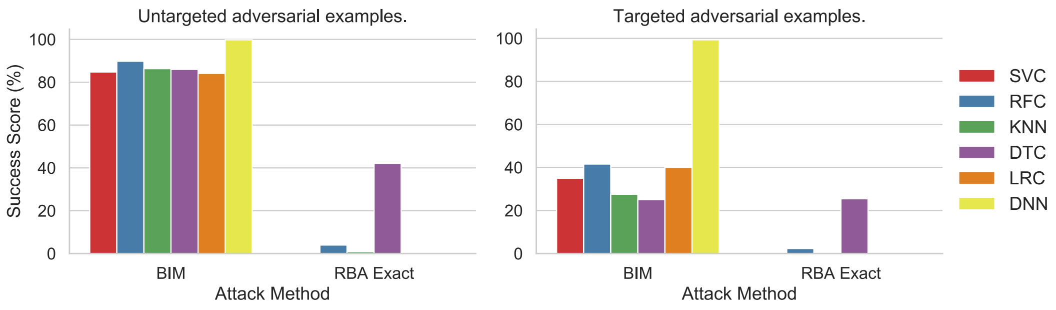


Fig. 11. Success score of adversarial examples computed using the DTC and DNN on different ML models.

the closest point in this region as an adversarial example. RBA is optimal and can find highly successful adversarial examples but suffer from high computational load. Heuristic decision tree attack searches for leaves in the decision tree with different class in the neighborhood of the leaf corresponding to the decision tree's original prediction for an input. The path from the original leaf to the adversarial leaf is used to modify the input sample to create an adversarial example. We used the Kernel Substitution Attack [28], which uses the FGSM, to craft adversarial examples misclassified by nearest neighbors with the KNN model.

Similar to the case of adversarial transferability between ML models, we trained six different ML algorithms on the feature data of the UCI dataset, and computed adversarial examples using DTC and KNN models. Figure 10, shows the success score of untargeted and targeted adversarial examples computed using the DTC model on all six ML models. Adversarial examples were able to fool the DTC model with good success score (40%), but performed poorly on other models, indicating poor adversarial transferability in both targeted and untargeted cases. We also want to highlight the difference in the success score of adversarial examples computed using the DNN model in Section 4 and adversarial examples computed using the DTC model here. Figure 11 shows the success score of adversarial examples computed using the DTC with RBA method and DNN model with BIM on all six ML models. As we can see, adversarial examples computed using the DNN model are more successful on the DTC, compared to adversarial examples computed using the DTC for the same adversarial perturbation budget. Furthermore, adversarial examples computed using the DNN model are more transferable than adversarial examples computed using the DTC.

Table 8 shows the success score of untargeted adversarial examples computed using the KNN model on all six ML models. The adversarial examples are highly transferable and the success scores are similar to those obtained with adversarial examples generated using the DNN as shown in Section 4. Therefore, KNN model is more vulnerable compared to the DTC model at the

Table 8. Success Score of Untargeted Adversarial Examples Computed Using the KNN Model on Different ML Models

	Machine Learning Models					
	SVC	RFC	KNN	DTC	LRC	DNN
Success Score (%)	85.74	89.82	86.01	85.32	86.40	77.35

same level of adversarial perturbation budget and adversarial examples computed using the KNN model is more transferable than DTC. However, adversarial attack methods that works with non-parametric learning algorithms such as DTC and KNN are much more computational intensive compared to gradient-based adversarial attack method and computed adversarial examples are also less successful and transferable. Hence, for sensor systems with computation and resource limitations, a direct attack on non-parametric learning algorithms might not be feasible. An attacker is better off using gradient-based attack methods to compute adversarial examples using the source systems and then attack the target system by exploiting the transferability of adversarial examples.

5.2 Feature Overlap

Authors in [17], have argued that neural networks trained on independent samples from a distribution tend to learn similar "*non-robust*" or brittle features making adversarial transferability possible. The central thesis is that data samples used to train ML model and used by an adversary belong to the same distribution. Therefore, models trained on similar data distributions have strong transferability between them, and models trained on distinct data distributions have weak transferability. This is because similar data distributions facilitate the learning of similar non-robust features and different data distribution has minimal overlap between the corresponding non-robust features. In adversarial transferability cases we have analyzed in this work, the data distribution of source and target systems have varying degree of overlap. In transferability between models, all models are trained on the same dataset. This allows the models to learn similar non-robust features resulting in excellent adversarial transferability. On the other hand, in transferability between datasets, source and target models are trained on datasets from different distributions. In this case, trained models have minimal overlap between learned non-robust features and consequently the adversarial transferability is poor. To verify this, we evaluated the target model on the test set of the source model. The performance of the target model on the source model's test set in theory is correlated with learned features shared between them. Higher classification accuracy of the target model on the source model test set implies learning of similar features, and lower classification accuracy demonstrates learning of different features between the source and the target model. The degree to which the target and source model share learned features is proportional to the performance of the target model on the adversarial examples computed using the source model.

Figure 12 shows the classification accuracy and success score of the target model on the source model test set and targeted adversarial examples computed using the source model with the Basic Iterative Attack (BIM). The performance of the target model on the source model test set was found to be directly proportional to the target model's success score on adversarial examples. Higher classification accuracy on the test set corresponded to a higher success score on adversarial examples and vice-versa. Hence, the degree to which features are shared between target and source models is directly related to the effectiveness of adversarial examples on the target model. Learning of similar features by target and source models facilitates better adversarial transferability, as demonstrated in transferability between models and transferability across subjects. On the

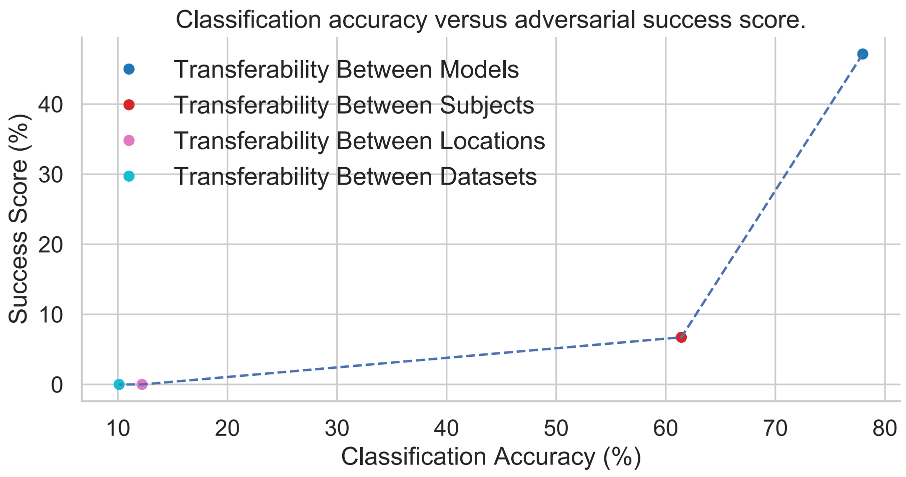


Fig. 12. The classification accuracy of the target model for different cases of transferability on the test set of the source model Vs. the success score of targeted adversarial examples computed using the BIM on the target model.

other hand, when target and source models have less overlap between learned features, the adversarial transferability is poor as found in transferability across sensor locations and transferability between datasets.

6 CONCLUSIONS

Adversarial examples are shown to be transferable across ML models trained on the whole or subset of the same dataset. However, the problem of adversarial transferability does not end there. For the first time in literature, we have investigated novel types of adversarial transferability in the context of wearable sensor systems with an extensive set of experiments. These new aspects of adversarial transferability show how an adversary can exploit sensor systems properties to craft adversarial examples in ways not discussed before. Our results not only demonstrate that there exist many new types of adversarial transferability but also show where and how these newer modes of transferability excel and fail.

We first evaluated the general case of transferability between ML models. Using the feature data from three real-world datasets, we found high untargeted transferability between different types of ML models with five attack methods. For targeted attacks, adversarial examples were less transferable for all three datasets with highest success score of 76%. We also found non-parametric learning algorithms such as DTC and KNNs to be more robust compared to other types of learning algorithms. The level of targeted transferability differed greatly across the three datasets. For Daily-Living (DL) dataset, both targeted and untargeted adversarial examples were much less likely to be transferable compared to UCI and MHEALTH datasets. The underlying reasons behind the low adversarial transferability for the DL dataset is due to the large number of samples, data collection in daily living condition, and no preprocessing steps for sensor data. These properties make adversarial examples less transferable because different learning algorithms learn different mappings between input and outputs.

For cross user transferability, we randomly selected data from half the subjects to create the source dataset and the data from the remaining half subjects was used as the target dataset. We separated all three datasets in this way and evaluated adversarial transferability across users. We discovered that the level of generalization between the source and target systems greatly affected the transferability of adversarial examples. For UCI and MHEALTH datasets, we found high level of generalization as indicated by the high classification accuracy of the source model on the target dataset and the target model on the source data. Consequently untargeted adversarial examples were highly transferable. For the DL dataset, we found low level of generalization between the source and target systems and as a result low levels of untargeted adversarial transferability. Targeted transferability was low for all three datasets and for the DL dataset success score of targeted adversarial examples was 0% at all levels of perturbation budget. Low targeted adversarial transferability performance indicates that the individual characteristics of users have significant bearings on adversarial transferability. Personalizing a model for a user will not only make the model achieve high performance on the user data but also make the model secure against adversarial transferability attacks.

Next, we evaluated the transferability across sensor body positions using the data from sensors placed at chest, left-ankle, and right-wrist. The adversarial transferability differed greatly based on source and target systems sensor body locations. For source and target systems sensors that were located near to each other, for examples right-wrist and chest, adversarial examples generated using the source system were highly transferable to the target system for both targeted and untargeted attacks. But for source and target systems sensor that were placed far-part, for example left-ankle and chest, the transferability of targeted adversarial examples was low. In the last experiment, we analyzed transferability across datasets. All three datasets have some common activity classes between them and some unique activity classes. For different combinations of source and target datasets, we found very low untargeted and targeted adversarial transferability for the entire spectrum of analysis.

In this work, we explored novel directions of adversarial transferability in the context of wearable sensor systems and showed how an adversary's performance varies in different transferability settings. In general untargeted attacks were more successful than targeted attacks. Untargeted attacks are considered successful as long as they can achieve random misclassification, which is much easier to achieve. The nature of the time-series input to sensor systems makes them more vulnerable to random misclassification because the data they operate on have properties that are easier to exploit. However, the complexity of adversarial attacks increases significantly in the targeted case. For targeted attacks, the attack methods have to find adversarial perturbations that need to conform to the temporal and spatial properties present in the dataset for the chosen target class. Due to this, the targeted transferability was very poor in most cases. Also, the main requirement of adversarial transferability is to have shared knowledge between the source and target systems. Irrespective of the learning algorithm, shared learning is facilitated when source and target datasets overlap along some dimensions such as processing routines, sensor type, sensor location, and population group. Hence, for an adversary to be successful it should take into consideration common attributes between the source and target systems and create adversarial examples that exploit these common attributes to have a higher chance of fooling the target system. In particular, our findings can be summarized as follows.

(1) The traditional notion of transferability across ML models showed excellent results which is consistent with the literature. But we also discovered that the properties of the underlying data distribution and properties of sensor system design such as the number of samples, context (in-lab or real-world) in which the dataset was collected, and preprocessing steps greatly affects adversarial transferability.
(2) We found gradient-based attack methods to be more competent at finding transferable adversarial perturbations compared to non-gradient based methods. Non-parametric learning algorithms such as decision tree and KNN were more robust against targeted and untargeted adversarial examples computed using both gradient and non-gradient based

attack methods. Furthermore, adversarial examples computed using DNN were more successful on these algorithms than adversarial examples computed using them.

(3) Individual characteristics of users greatly affect targeted adversarial transferability. Also, if the source and target datasets are from the same population, then the sensor location of source and target systems becomes important. Near source and target sensor locations facilitate higher adversarial transferability and vice-versa.

(4) In general, the extent to which the source and target systems properties overlap affects adversarial transferability. The properties of source and target systems are mainly governed by source and target datasets and models. Datasets encode several attributes of wearable sensor systems such as sensor type, subjective biases, preprocessing pipeline, sensor placement and orientation. Models represent types of learning algorithms and attributes of the algorithm. Adversarial transferability depends on the shared knowledge between source and target systems, and depending on the extent to which both systems share common mapping between inputs and outputs adversarial transferability varies. In the case of transferability between models, the distinction between source and target systems is only for the learning algorithms and in this case we found high untargeted and targeted transferability. On the other hand, in the case of transferability between datasets, source and target systems learning algorithms are the same, but source and target datasets are different. In this case, we observed low transferability for both untargeted and targeted attacks. Therefore the main reason for adversarial transferability is the similarity between source and target datasets. By increasing the distance between source and target datasets using design principles or post-processing techniques, adversarial transferability can be significantly reduced.

7 RECOMMENDATIONS

In this section, we provide recommendations aimed at a system designer for designing robust embedded systems based on our results and findings. These recommendations can be considered as design choices that can affect the adversarial robustness of a sensor system.

(1) The fundamental reason for the robustness of the target system against adversarial transferability was the distance between the source and target data distributions. With the increasing level of distance between the source and target data distribution as shown in Section 5.1, the adversarial transferability decreased and reached a success score of 0%. Hence, when designing and developing embedded sensor systems it is recommended to use proprietary or private datasets. If it is not possible to use private datasets, data processing techniques such as Principal Component Analysis [6], removing non-robust samples from the dataset [38] and noise addition should be used as preprocessing steps on the dataset to learn a robust classifier.

(2) Personalizing ML models for a user, often needed in sensor system applications, shows the potential of not only improving the model performance for the user but also make the model robust against adversarial attacks.

(3) Sensor system trained on a large real-world dataset was discovered to be more robust to adversarial transferability compared to a system trained with smaller lab-setting datasets. In our analysis, target systems that used the Daily Living (DL) dataset (sample size 16,434) were more robust towards both untargeted and targeted adversarial examples than target systems that used the UCI dataset (sample size 10,299) and the MHEALTH dataset (sample 5,133). Hence, it is better to have a large dataset for a robust system from an adversarial transferability point of view.

Finally, we want to draw the reader's attention to the argument that ML systems can be protected by access control, and very few cases of adversarial attacks can happen in real-world wearable systems. However, by limiting our understanding of vulnerabilities that exists in sensor systems by operating on the default setting that adversarial attacks on embedded systems have a low chance of occurrence is not prudent. If we ignore the discussion of adversarial attacks and transferability by operating on the default setting, we will be blind to the inherent shortcomings of our systems, which can be detrimental to the overall health of our systems. For example, consider a fall detection system used to dispatch help when the system detects falls. If an adversary can influence any aspect of this system, then the effect can have life-altering consequences. Furthermore, recent works have shown that adversarial attacks are possible in real-world conditions, and the transferability of adversarial examples dramatically enhances the chances of success for an adversary [13, 15, 16, 23]. Also, the decision-making model needs not to be present locally on the device. The model can be in the cloud, and the system operates by querying the cloud model with sensor readings for classification [10, 32]. This mode of operation is becoming more mainstream as it provides many benefits, such as life-long learning, active learning, and data analytics. Therefore, acknowledging and understanding the adversarial nature of ML algorithms used in embedded sensor systems allow us to build measures and adapt the design process to thwart and limit the impact of adversarial attacks. This is precisely what we aimed to achieve in this work. By making the connection between adversarial transferability and different aspects of embedded sensor systems, we showed where the strengths and limitations of an adversary lie and how a system designer can use this information to design and build robust and reliable embedded systems.

8 LIMITATIONS AND FUTURE WORK

In this work, we have tried to cover the topic of adversarial transferability in embedded systems in a broad manner. Nonetheless, our work does have some limitations, which we have highlighted below.

- In our experiments, we have used five different adversarial attack methods to evaluate adversarial transferability in embedded sensor systems. However, there are many more attack methods in the literature that we have left out of our discussion. Unexplored attack methods with better optimization strategies may be able to find adversarial perturbation with better transferability properties and succeed where the discussed attack methods have failed.
- The discussion of adversarial ML is not complete without talking about defense against adversarial attacks. Attack and defense form the two faces of the adversarial ML coin, and hence should be given equal importance and attention in research. Our discussion in this work does not discuss defense mechanisms, and we aim to explore the effects of defense methods against adversarial transferability in our future works.
- In this work, we have only discussed the level of performance of different attack methods in terms of transferability. One interesting question that we can ask based on our results is, "What makes some attack methods to have higher or lower rates transferability than others?". This is one of the fundamental questions that need to be investigated to better understand the results obtained in this work.

Finally, we want to touch upon the indistinguishability of signals and the requirement of adversarial perturbation budget in the case of sensor systems. We know that adversarial examples are computationally created inputs not significantly different from samples in the target data distribution. However, signals lack the observational understanding present in samples from domain such as computer vision. It is difficult and almost impossible to understand a signal by observation without some operation to quantify its properties. Hence, in signal domain adversarial examples extends the traditional definition and encompasses a broad spectrum of generation schemes. For example, an attacker can send any signal conforming to the characteristics of the target class without any other consideration and any good target model will be fooled. Also, the lack of understanding of signals makes it almost impossible to determine whether an input signal is adversarial or not just by observation without knowing the actual ground truth label. Perturbation budget also plays a role in defining the extent to which an adversarial example can differ from actual samples from the data distribution. Therefore, in signal domain, the requirements of perturbation budget need further analysis.

APPENDICES

A DATASETS ACTIVITY DISTRIBUTION

Figure 13 shows the class distribution of all three datasets used in the analyses. The class distribution is fairly balanced for all three datasets.

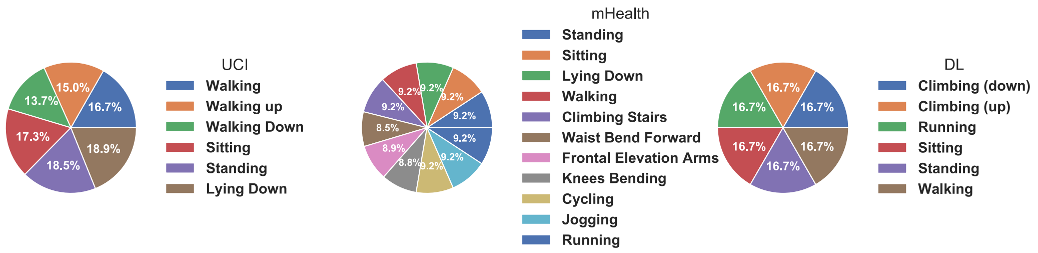
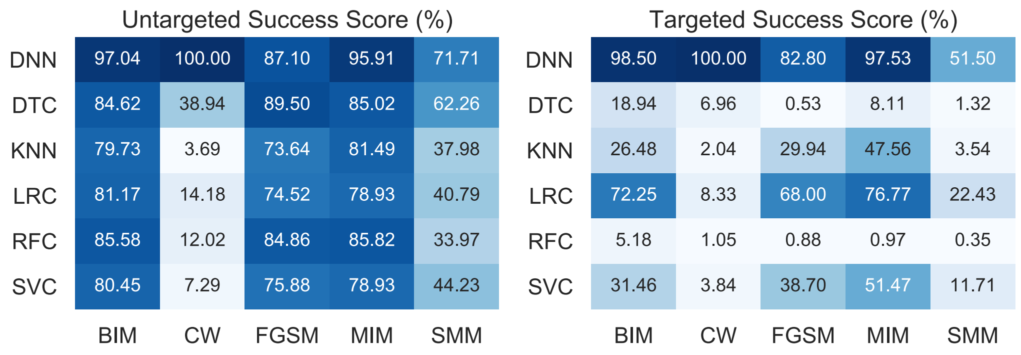


Fig. 13. Activity distribution of the three datasets (best viewed in color).

B TRANSFERABILITY BETWEEN ML MODELS—MHEALTH DATASET

Figure 14 shows the success scores of untargeted and targeted transferability between models for the MHEALTH dataset. We found high level of untargeted transferability with all attack methods except the CW attack. Also, targeted transferability was high reaching success scores up to 76.77%.



Untargeted Success Score (%)

	BIM	CW	FGSM	MIM	SMM
DNN	97.04	100.00	87.10	95.91	71.71
DTC	84.62	38.94	89.50	85.02	62.26
KNN	79.73	3.69	73.64	81.49	37.98
LRC	81.17	14.18	74.52	78.93	40.79
RFC	85.58	12.02	84.86	85.82	33.97
SVC	80.45	7.29	75.88	78.93	44.23

Targeted Success Score (%)

	BIM	CW	FGSM	MIM	SMM
DNN	98.50	100.00	82.80	97.53	51.50
DTC	18.94	6.96	0.53	8.11	1.32
KNN	26.48	2.04	29.94	47.56	3.54
LRC	72.25	8.33	68.00	76.77	22.43
RFC	5.18	1.05	0.88	0.97	0.35
SVC	31.46	3.84	38.70	51.47	11.71

Fig. 14. Success score of untargeted and targeted adversarial examples for the MHEALTH dataset.

C TRANSFERABILITY ACROSS USERS—MHEALTH DATASET

Figure 15 shows the success scores of untargeted and targeted transferability across users for the MHEALTH dataset. The activity class "*Walking*" was used as the target class. We found high level of untargeted transferability, but targeted transferability was low at all level of perturbation budgets.

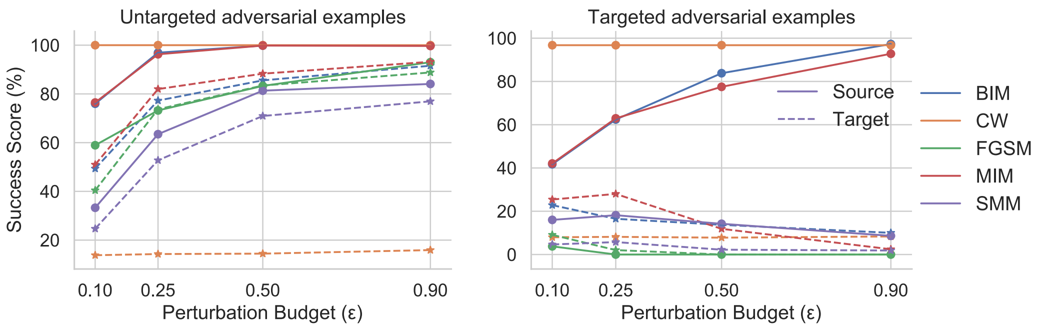


Fig. 15. Success score of adversarial examples on source and target systems with the MHEALTH dataset for transferability across users.

D TRANSFERABILITY ACROSS SENSOR BODY LOCATIONS—LEFT-ANKLE VS. RIGHT-WRIST

Figure 16 shows the success score of untargeted and targeted adversarial examples computed using the left-ankle (source) system on the left-ankle and right-wrist (target) systems. Similar to Chest Vs. Left-Ankle case, we found high untargeted transferability, success score upto 98%, and very low (0%) targeted transferability for the target class of "Sitting".

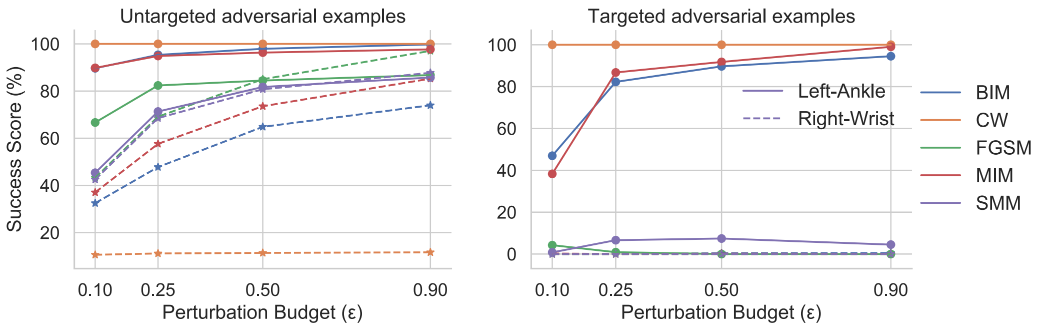


Fig. 16. Success score of adversarial examples computed using the source system (**Left-Ankle**) on source and target (**Right-Wrist**) systems.

E TRANSFERABILITY BETWEEN DATASETS—DL VS. MHEALTH

Tables 9 and 10 show untargeted and targeted transferability for the case of source MHEALTH dataset and target DL dataset. Similar to UCI Vs. MHEALTH case, we found good level of untargeted transferability but no targeted transferability at all of perturbation budgets.

Table 9. Success Score of Untargeted Adversarial Examples Computed Using the Source (MHEALTH) System on the Source and Target (DL) Systems

Evasion Attack Methods	Untargeted Attack Perturbation Budget (ϵ)							
	0.1		0.25		0.5		0.9	
	Source	Target	Source	Target	Source	Target	Source	Target
FGSM	64.58	24.35	83.33	30.84	90.30	30.60	92.54	31.25
BIM	80.76	21.47	99.83	33.89	99.91	36.21	100.0	38.46
MIM	82.69	25.56	99.43	32.37	99.91	35.25	100.0	39.98
SMM	40.46	20.99	67.06	35.33	81.00	38.38	86.21	38.46
CW	100.0	12.58	100.0	12.41	100.0	12.66	100.0	13.14

Table 10. Success Score of Targeted Adversarial Examples Computed Using the Source (MHEALTH) System on the Source and Target (DL) Systems

Evasion Attack Methods	Targeted Attack Perturbation Budget (ϵ)							
	0.1		0.25		0.5		0.9	
	Source	Target	Source	Target	Source	Target	Source	Target
FGSM	0.70	0.0	0.26	0.0	0.0	0.0	0.0	0.0
BIM	23.43	0.0	68.16	0.0	88.85	0.0	98.14	0.0
MIM	30.50	0.0	64.98	0.0	86.38	0.0	93.81	0.0
SMM	17.41	0.0	24.49	0.0	15.29	0.0	5.39	0.0
CW	100.0	0.0	100.00	0.0	100.0	0.0	100.0	0.0

F MANIFOLD LEARNING

Manifold learning methods seek to describe high-dimensional data in low dimensional space. We used ***Multidimensional Scaling*** (**MDS**) [19] to generate low-dimensional representations of adversarial examples and clean samples. MDS uses a pair-wise distance matrix as inputs and places each data point in an n-dimensional space such that the distance between the points is preserved as well as possible. Albeit, the Euclidean distance suffers from the curse of dimensionality when used to compute the distance between objects in high-dimensional space, it can still be used to compute the similarity matrix between adversarial and clean samples. This similarity matrix is used by multidimensional scaling to get the low-dimensional representation of the adversarial and clean samples. To compute the low dimensional embedding, we used all samples from the target model's training set for the target class and the top-k samples from the targeted adversarial set that was classified into the target class by the target model. Here, k is the number of samples selected from the target model training set, and we sort the prediction confidence of the adversarial examples for the target class to determine the top-k examples. In cases where adversarial examples fail to fool the target model, we take k random samples from the adversarial set.

Figure 17 shows the multidimensional scaling of adversarial examples and benign samples from the target model's training set for different cases of adversarial transferability. For transferability between models, the 2-dimensional representation of clean and adversarial samples share a significant overlap region, which corresponds to the high targeted transferability we observed in this case. The region of overlap for transferability across subjects is not significant, but the spatial distribution of adversarial and benign samples share shape and organization, which demonstrate the fair transferability in this case. For transferability across sensor body locations and datasets, the representation of benign and adversarial samples shares neither region nor organization and consequently we observed poor targeted adversarial transferability for these cases in our results. The representation of clean and adversarial samples in 2−dimensional space gives us insights about the

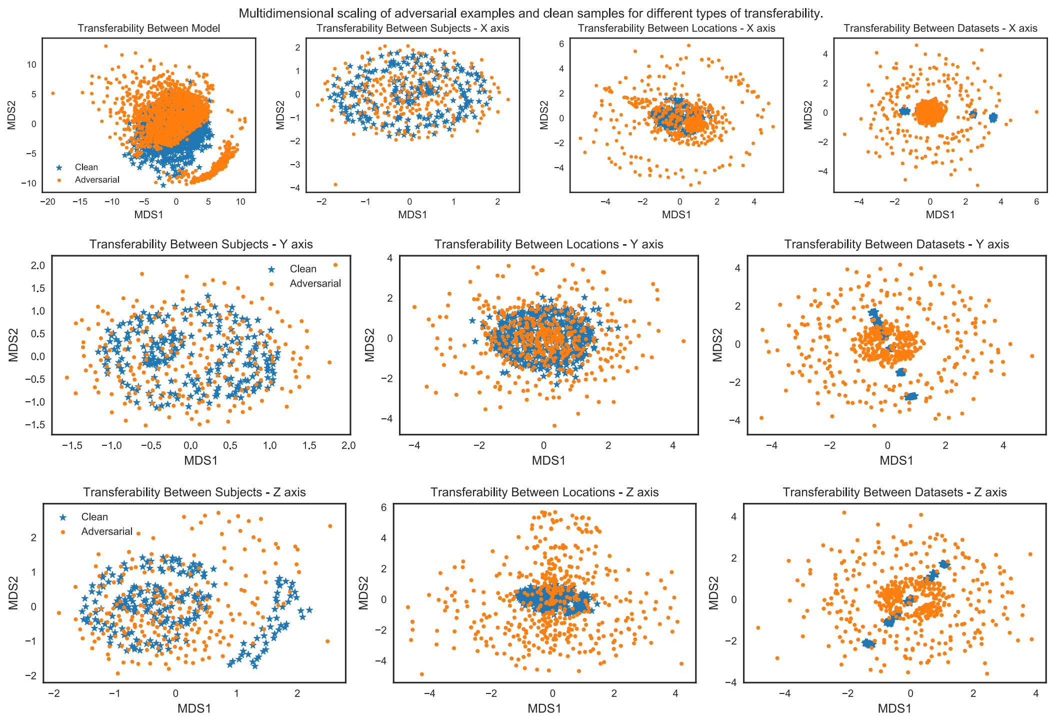


Fig. 17. MDS of clean and adversarial samples for different cases of targeted adversarial transferability. Except for transferability between models which was evaluated with 1D feature data, other types of adversarial transferability uses 3−axial accelerometer data and hence we have plots of Y axis and Z axis for these modes of adversarial transferability.

transferability results we obtained in our experiments. Our aim here was to demonstrate how the spatial distribution of adversarial and benign samples looks like for different cases of adversarial transferability and explain the results we obtained from our experiments. The degree to which adversarial samples can conform to benign samples from the target model's training set is directly proportional to adversarial examples success score on the target system.

G PERTURBATION SIZE AND ATTACK METHODS

Given that data in our experiments are scaled in the range $[1, -1]$ and we have used adversarial perturbation budgets up to 0.9, it is natural to assume that adversarial examples computed at higher perturbation budgets will be significantly different from benign samples used to create such adversarial examples. However, this is only true for the FGSM attack method because FGSM uses l-∞ norm and every entry in the input vector can be modified by half the perturbation budget value. Other attack methods, for example BIM which is an iterative version of FGSM, behaves in a different way and find adversarial perturbation which are limited to perturbation budget allowed for each iteration. In our experiments, the number of iteration is set to 50, and consequently the perturbation budget for all iterative methods per iteration will be $0.9/50 = 0.018$. We have presented a visual demonstration in Figures 18 and 19. We computed untargeted and targeted adversarial examples using the FGSM and BIM attacks at different level of adversarial perturbation budget. As expected, adversarial examples computed using the FGSM is very different compared to the input and at higher perturbation budget this difference is significant. On the other hand, adversarial

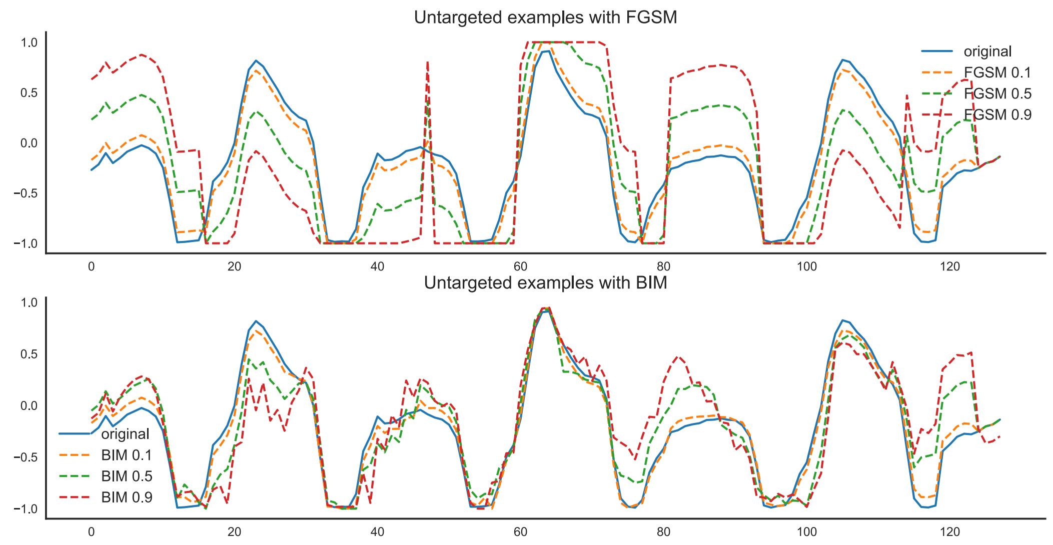


Fig. 18. Untargeted adversarial examples computed at different perturbation budgets with FGSM and BIM attacks. Image best viewed in color.

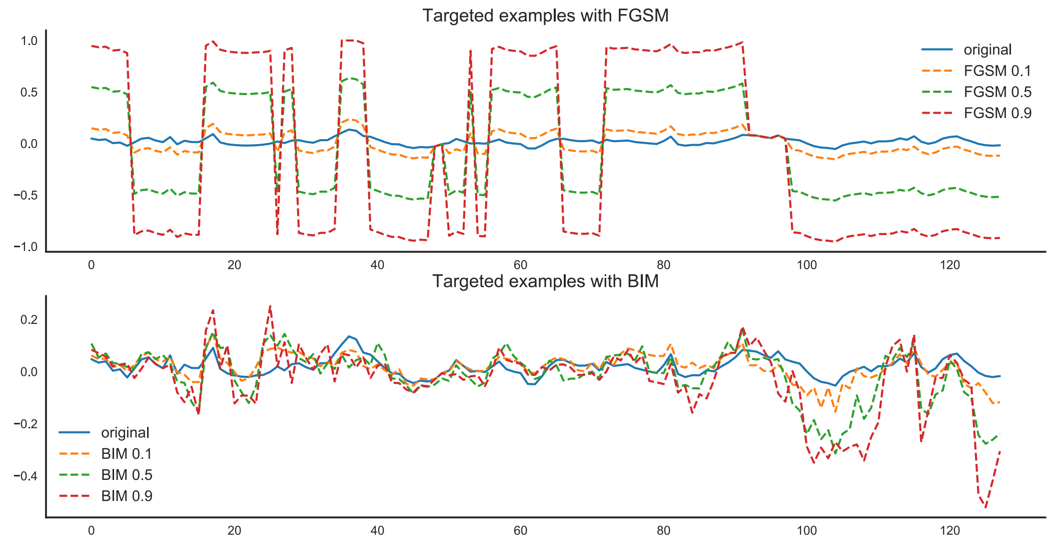


Fig. 19. Targeted adversarial examples computed at different perturbation budgets with FGSM and BIM attacks. Image best viewed in color.

examples computed using the BIM is very similar to the input even at high level of perturbation budgets.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, PeteWarden,

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: a system for large-scale machine learning. *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation (OSDI'16, Savannah, GA, USA)*, USENIX Association, 265–283.

[2] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. 2013. A public domain dataset for human activity recognition using smartphones. *The European Symposium on Artificial Neural Networks.* https://api.semanticscholar.org/CorpusID:6975432

[3] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. 2018. Synthesizing robust adversarial examples. In *Proceedings of the International Conference on Machine Learning*. 284–293.

[4] Oresti Banos, Juan-Manuel Galvez, Miguel Damas, Hector Pomares, and Ignacio Rojas. 2014. Window size impact in human activity recognition. *Sensors* 14, 4 (2014), 6474–6499. Retrieved from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4029702/

[5] Oresti Banos, Claudia Villalonga, Rafael Garcia, Alejandro Saez, Miguel Damas, Juan A Holgado-Terriza, Sungyong Lee, Hector Pomares, and Ignacio Rojas. 2015. Design, implementation and validation of a novel open framework for agile development of mobile health applications. *Biomedical Engineering Online* 14, 2 (2015), S6.

[6] Arjun Nitin Bhagoji, Daniel Cullina, Chawin Sitawarin, and Prateek Mittal. 2017. Enhancing robustness of machine learning systems via data transformations. *52nd Annual Conference on Information Sciences and Systems (CISS'18)*, 1–5. https://api.semanticscholar.org/CorpusID:37108626

[7] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Srndic, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. 2013. Evasion attacks against machine learning at test time. arXiv:1708.06131. Retrieved from https://arxiv.org/abs/1708.06131

[8] Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *Proceedings of the 2017 IEEE Symposium on Security and Privacy*. IEEE, 39–57.

[9] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. 2018. Adversarial attacks and defences: A survey. arXiv:1810.00069. Retrieved from https://arxiv.org/abs/1810.00069

[10] Nicolas Escobar Cruz, Jhon Solarte, and Andres Gonzalez-Vargas. 2018. Automated epileptic seizure detection system based on a wearable prototype and cloud computing to assist people with epilepsy. In *Proceedings of the Workshop on Engineering Applications*. Springer, 204–213.

[11] Ambra Demontis, Marco Melis, Maura Pintor, Matthew Jagielski, Battista Biggio, Alina Oprea, Cristina Nita-Rotaru, and Fabio Roli. 2019. Why do adversarial attacks transfer? Explaining transferability of evasion and poisoning attacks. In *Proceedings of the 28th USENIX Conference on Security Symposium (SEC'19)*. USENIX Association, Berkeley, CA, USA, 321–338. Retrieved from http://dl.acm.org/citation.cfm?id=3361338.3361361

[12] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. 2018. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 9185–9193.

[13] Yuan Gong, Boyang Li, Christian Poellabauer, and Yiyu Shi. 2019. Real-time adversarial attacks. arXiv:1905.13399. Retrieved from https://arxiv.org/abs/1905.13399

[14] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. arXiv:1412.6572. Retrieved from https://arxiv.org/abs/1412.6572

[15] Dou Goodman. 2020. Transferability of adversarial examples to attack cloud-based image classifier service. arXiv:2001.03460. Retrieved from https://arxiv.org/abs/2001.03460

[16] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. 2018. Black-box adversarial attacks with limited queries and information. *International Conference on Machine Learning*. https://api.semanticscholar.org/CorpusID:5046541

[17] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. 2019. Adversarial examples are not bugs, they are features. In *Proceedings of the Advances in Neural Information Processing Systems 32.* H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Curran Associates, Inc., 125–136. Retrieved from http://papers.nips.cc/paper/8307-adversarial-examples-are-not-bugs-they-are-features.pdf

[18] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. arXiv:1412.6980. Retrieved from https://api.semanticscholar.org/CorpusID:6628106

[19] Joseph Kruskal and Myron Wish. 2020. *Multidimensional Scaling*. Thousand Oaks, California. DOI: https://doi.org/10.4135/9781412985130

[20] Mukesh Kumar and S. L. Shimi. 2015. Voice recognition based home automation system for paralyzed people. *International Journal of Advanced Research in Electronics and Communication Engineering* 4, 10 (2015), 671–673.

[21] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. 2018. Adversarial examples in the physical world. In *Proceedings of the Artificial Intelligence Safety and Security*. Chapman and Hall/CRC, 99–112.

[22] Oscar D. Lara and Miguel A. Labrador. 2013. A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys and Tutorial* 15 (2013), 1192–1209. Retrieved from http://ieeexplore.ieee.org/document/6365160/

[23] Zhuohang Li, Cong Shi, Yi Xie, Jian Liu, Bo Yuan, and Yingying Chen. 2020. Practical adversarial attacks against speaker recognition systems. In *Proceedings of the 21st International Workshop on Mobile Computing Systems and Applications (HotMobile'20)*. Association for Computing Machinery, New York, NY, USA, 9–14. DOI: https://doi.org/10.1145/3376897.3377856

[24] U. Maurer, A. Smailagic, D. P. Siewiorek, and M. Deisher. 2006. Activity recognition and monitoring using multiple sensors on different body positions. In *Proceedings of the International Workshop on Wearable and Implantable Body Sensor Networks (BSN'06)*. DOI: https://doi.org/10.1109/BSN.2006.6

[25] Muzammal Naseer, Salman H. Khan, Harris Khan, Fahad Shahbaz Khan, and Fatih Porikli. 2019. Cross-domain transferability of adversarial perturbations. *Neural Information Processing Systems*. https://api.semanticscholar.org/CorpusID:167217657

[26] Tsuyoshi Okita and Sozo Inoue. 2018. Activity recognition: Translation across sensor modalities using deep learning. In *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers (UbiComp'18)*. Association for Computing Machinery, New York, NY, USA, 1462–1471. DOI: https://doi.org/10.1145/3267305.3267512

[27] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, Rujun Long, and Patrick Mcdaniel. 2018. Technical report on the cleverhans v2.1.0 adversarial examples library. arXiv:1610.00768. Retrieved from https://arxiv.org/abs/1610.00768

[28] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. 2016. Practical black-box attacks against machine learning. *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. https://api.semanticscholar.org/CorpusID:1090603

[29] Nicolas Papernot, Patrick McDaniel, and Ian J. Goodfellow. 2016. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv:1605.07277. Retrieved from https://arxiv.org/abs/1605.07277

[30] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. 2015. The limitations of deep learning in adversarial settings. *IEEE European Symposium on Security and Privacy (EuroS&P'16)*, 372–387. https://api.semanticscholar.org/CorpusID:7004303

[31] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research* 12, 85 (2011), 2825–2830. http://jmlr.org/papers/v12/pedregosa11a.html

[32] L. Rachakonda, P. Sundaravadivel, S. P. Mohanty, E. Kougianos, and M. Ganapathiraju. 2018. A smart sensor in the IoMT for stress level detection. In *Proceedings of the 2018 IEEE International Symposium on Smart Electronic Systems*. 141–145.

[33] Ramesh K. Sah and Hassan Ghasemzadeh. 2019. Adar: Adversarial activity recognition in wearables. In *Proceedings of the 2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*.

[34] Muhammad Shoaib, Hans Scholten, and Paul J. M. Havinga. 2013. Towards physical activity recognition using smartphone sensors. In *Proceedings of the 2013 IEEE 10th International Conference on Ubiquitous Intelligence and Computing and 2013 IEEE 10th International Conference on Autonomic and Trusted Computing*. IEEE, 80–87.

[35] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *Proceedings of the International Conference on Learning Representations*.

[36] Timo Sztyler, Johanna Völker, Josep Carmona Vargas, Oliver Meier, and Heiner Stuckenschmidt. 2015. Discovery of personal processes from labeled sensor data: An application of process mining to personalized health care. In *Proceedings of the International Workshop on Algorithms and Theories for the Analysis of Event Data: Brussels, Belgium, June 22–23, 2015*. CEUR-WS. org, 31–46.

[37] Yi Yang, Hengliang Luo, Huarong Xu, and Fuchao Wu. 2015. Towards real-time traffic sign detection and classification. *IEEE Transactions on Intelligent Transportation Systems* 17, 7 (2015), 2022–2031.

[38] Yao-Yuan Yang, Cyrus Rashtchian, Yizhen Wang, and Kamalika Chaudhuri. 2020. Robustness for non-parametric classification: A generic attack and defense. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*. PMLR, 941–951.

[39] X. Yuan, P. He, Q. Zhu, and X. Li. 2019. Adversarial examples: Attacks and defenses for deep learning. *IEEE Transactions on Neural Networks and Learning Systems* 30, 9 (2019), 2805–2824. DOI: https://doi.org/10.1109/TNNLS.2018.2886017

[40] Mi Zhang and Alexander A. Sawchuk. 2011. A feature selection-based framework for human activity recognition using wearable multimodal sensors. In *Proceedings of the BodyNets.* 92–98.

[41] Vincent Wenchen Zheng, Derek Hao Hu, and Qiang Yang. 2009. Cross-domain activity recognition. In *Proceedings of the 11th International Conference on Ubiquitous Computing (UbiComp'09).* Association for Computing Machinery, New York, NY, USA, 61–70. DOI: https://doi.org/10.1145/1620545.1620554

Received 1 February 2022; revised 8 November 2023; accepted 15 January 2024