Hand Me Your PIN!
Inferring ATM PINs of Users Typing with a Covered Hand

Matteo Cardaioli
University of Padua, Italy
GFT Italia, Italy

Stefano Cecconello
University of Padua, Italy

Mauro Conti
University of Padua, Italy

Simone Milani
University of Padua, Italy

Stjepan Picek
Delft University of Technology, The Netherlands

Eugen Saraci
University of Padua, Italy

Abstract

Automated Teller Machines (ATMs) represent the most used system for withdrawing cash. The European Central Bank reported more than 11 billion cash withdrawals and loading/unloading transactions on the European ATMs in 2019. Although ATMs have undergone various technological evolutions, Personal Identification Numbers (PINs) are still the most common authentication method for these devices. Unfortunately, the PIN mechanism is vulnerable to shoulder-surfing attacks performed via hidden cameras installed near the ATM to catch the PIN pad. To overcome this problem, people get used to covering the typing hand with the other hand. While such users probably believe this behavior is safe enough to protect against mentioned attacks, there is no clear assessment of this countermeasure in the scientific literature.

This paper proposes a novel attack to reconstruct PINs entered by victims covering the typing hand with the other hand. We consider the setting where the attacker can access an ATM PIN pad of the same brand/model as the target one. Afterward, the attacker uses that model to infer the digits pressed by the victim while entering the PIN. Our attack owes its success to a carefully selected deep learning architecture that can infer the PIN from the typing hand position and movements. We run a detailed experimental analysis including 58 users. With our approach, we can guess 30% of the 5-digit PINs within three attempts – the ones usually allowed by ATM before blocking the card. We also conducted a survey with 78 users that managed to reach an accuracy of only 7.92% on average for the same setting. Finally, we evaluate a shielding countermeasure that proved to be rather inefficient unless the whole keypad is shielded.

1 Introduction

The wide deployment of various Cyber-Physical Systems (CPS) has a significant impact on our daily lives. Unfortunately, the increased use of CPS also brings more threats to users. This is especially pronounced considering new attack vectors that use machine learning approaches. As such, threats become a global issue, and the need to design secure and robust systems increases. One common security mechanism in devices like Automated Teller Machines (ATMs) and Point of Sale (PoS) depends on the security provided by the Personal Identification Numbers (PINs). While ATMs and PoS devices are widely used [1], many people do not consider security risks and defenses beyond those commonly mentioned [2]: i) hide the PIN while typing, and i) make sure no one watches the screen (shoulder-surfing attack). In the context of financial services, ISO 9564-1 [18] specifies the basic security principles for PINs and PIN entry devices (e.g., PIN pads). For example, to mitigate the shoulder surfing attacks [5, 12], the standard indicates that i) PIN digits must not be displayed on a screen, and ii) the duration and type of feedback sound emitted must be the same for each key. Consequently, as a compromise between security and usability, PIN entry systems display a fixed symbol (e.g., a dot) to represent a digit being pressed and provide the same audio feedback (i.e., same tone, same duration) for all keys. Thus, the combination of security mechanisms enforced by standards and the common precaution measures taken by users should provide sufficient protection. Unfortunately, the attackers also improve their approaches over time and consider more sophisticated attacks.

The security of ATM and PoS devices is of great concern as millions of such devices are used [10]. Resourceful attackers that succeed in attacking even a small percentage of those devices can cause significant damage considering costs and public perception. This problem is especially pronounced as last years brought significant developments in the attack techniques [2, 8, 25]. At the same time, attacking ATM or PoS devices is not easy, especially if considering realistic settings. Most of the state-of-the-art attacks can be defeated by a careful user covering the PIN that is entered. Recent results that consider thermal cameras are also difficult to succeed, depending on the keypad type and the time users spend operating the device. The attacker can also use timing or acoustic

[1] https://sdw.ecb.europa.eu/reports.do?node=1000001407
[2] https://www.hsbc.com.hk/help/cybersecurity-and-fraud/atm-scams/

attacks to infer information about the entered digits, but they are not as effective as the state-of-the-art attacks since they require additional information such as thermal residues [8], making it challenging to apply realistically such attacks.

This work proposes a novel attack aiming to reconstruct PINs entered by victims that cover the typing hand by the other hand. More precisely, we leverage the advances in the deep learning domain to develop an attack predicting what PIN is entered based on the position of the user's hand and the movements while pressing the keys. Our attack gives high accuracy rates even in the cases when the user perfectly covers the typing hand. What is more, our attack reaches higher accuracy values than previous works that needed to consider several sources of the information at the same time (timing, sound, and thermal signatures) [8].

Our attack considers a profiling setting where the attacker has access to a PIN pad that is identical (or at least similar) to the one used by the victim. Then, we build a profiling model that can predict what digit is entered on the target device. This is the first attack on PIN mechanisms that works even when the PIN is covered while being entered to the best of our knowledge. Our attack demonstrates that the ATM and PoS security mechanisms are insufficient, and we must provide novel defenses to mitigate attackers. We made our code and datasets publicly available at https://spritz.math.unipd.it/projects/HandMeYourPIN.

Main contributions

- We propose a novel attack to infer PINs from videos of users covering the typing hand with their non-typing hand.
- We demonstrate that our attack can reconstruct 30% of 5-digit PINs and 41% of 4-digit PINs within three attempts, showing that hiding the PIN while typing is insufficient to ensure proper protection.
- We evaluate our attack via extensive experiments, collecting videos of 5 800 5-digit PINs entered in a simulated ATM by 58 participants. We conduct a study to assess humans' accuracy in inferring covered PINs from videos. We show that our attack outperforms humans, achieving a four-fold improvement on reconstructing 5-digits PINs within three attempts.
- We pre-process our dataset, and we make it publicly available to the research community. We hope this is beneficial to understand the problem better and propose possible solutions.
- We discuss several countermeasures that would make the attack more difficult to conduct. We perform an analysis on the attack performance when covering the PIN pad (coverage 25%, 50%, 75%, and 100%) and show that attacks are possible even when using this countermeasure.

2 Threat Model

The attack is performed when a victim interacts with a generic ATM keypad and types the PIN. The ATM is equipped with a PIN pad that emits a feedback sound when a key is pressed. The feedback sound is the same for all the keys of the PIN pad. The ATM is equipped with a monitor where obfuscated symbols appear when users enter a PIN to mask the entered digits. We do not assume that the ATM or its PIN pad have been compromised during the attack. Our approach can be considered an alternative to card-skimmer attacks since we consider a different source of information to retrieve the PIN. Usually, card-skimming attacks rely on fake PIN pads that directly record the entered digits [30], while our approach infers the PINs from a video.

2.1 Attacker

The attacker is a malicious user aiming to steal the victim's secret PIN. The attacker can place a hidden camera near the ATM to record the PIN pad. We make no assumptions about the type of camera used by the attacker except that it records in the visible spectrum [3]. We assume that the camera can easily be hidden close to the ATM while keeping a direct view of the PIN pad (i.e., a pinhole camera if the attacker has access to the ATM [4] or any standard camera placed outside the ATM chassis). We also do not assume any specific position for the camera, but we discuss various camera placements' advantages. We primarily consider the scenario where the attacker uses only one camera, but we also discuss the attack performance when using multiple cameras.

The attack may take place together with different card stealing approaches: i) card skimming both on chip [7] or magnetic stripe [30] (currently, the two payment-enabling technologies work together [9]), ii) exploiting a relay attack on a contactless card [14], and iii) physically stealing the victim's card.

We assume a profiling side-channel attack where the side-channel information comes from the video of the victim's hand while entering the PIN. More precisely, side-channel information is the position of the victim's hand and the hand movements (both moving the hand/fingers to reach different keypads or movements observable due to muscle movements while a certain keypad is pressed). The attacker can record a number of PINs entered on a copy of the ATM device and train a profiling model to predict what key is pressed. The attacker can retrieve the timestamps when the victim has typed the single keys on the keypad and can do so by listening to the audio of the video recording. There are two different types of sound clues that the attacker can exploit: the first one is the feedback sound made by the keypad when a key is pressed [8], the second one is the sound of the physical button

[3] We will use cheap and easily-concealable video sensing equipment, where standard RGB cameras fit such requirements.

[4] https://www.sperrywest.com/cameras/

of the keypad that is pressed. External noise does not prevent the attacker from extracting the keypresses, as the camera is close enough to the keypad. As such, the sound can still be identified in the audio track. If, for any reason, the attacker has no way to retrieve the timestamps from the recorded audio (or if there is no audio at all), it is possible to place the camera to record both the keypad and the screen of the ATM [2]. This allows the attacker to extract the keypresses' timing by looking at the PIN masking symbols appearing on the screen. Common masking symbols are usually dots and asterisks. The attacker can use any method to build a profiling model to predict what keys are pressed. We consider the top three predictions as a measure of success since most ATMs will allow entering the PIN three times before blocking the card. Finally, we do not assume that the PIN has any specific structure (pattern) that could be used to improve the attack performance further.

2.2 Victim

We assume that the victim adopts basic countermeasures against card-skimming attacks, such as covering the hand while entering the PIN. The attacker does not need to be there when the victim types the PIN, as the attacker can freely access the camera's recorded video, either remotely or at a different time.

3 Attack Approach

Our attack assumes that the attacker has access to a training device and controls the PIN selection. Additionally, the attacker knows the layout of a target device and will select the training device to be similar. The attacker does not know the specific person to be attacked or the PIN for the attacked device.

3.1 Attack Phases

We can divide the attack into three phases: Phase A – Training, Phase B – Video Recording, and Phase C – PIN Inference. Figure 1 shows the required steps for the attack.

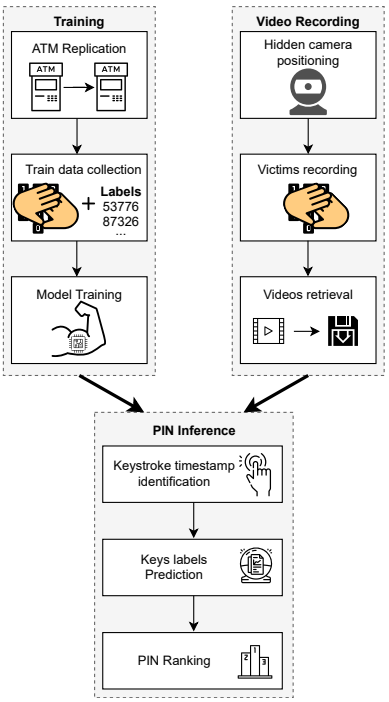


Figure 1: The attack step-by-step. The data collection process does not necessarily need to happen before the attacker steals the victim's PIN. Still, it is a required step of the attack.

Phase A – Training

The attacker selects an ATM as the target of the attack. Next, the attacker sets up a replica of the target ATM. This replica does not have to be a faithful copy of the original, as our model takes in as input a crop around the keypad of the ATM. Therefore, the attacker must use a keypad similar to the one on the target ATM. The best situation is when the attacker can retrieve the same PIN pad model. Alternatively, the attacker can also use PIN pads that differ slightly (e.g., the key spacing can vary by a few millimeters). Note that the layout of ATM PIN pads has to follow the ISO 9564 standard [18]. The attacker uses the ATM replica to build the training set, simulating the victim's behavior while entering the PIN (i.e., covering the typing hand). The attacker must enter sequences of PINs on the replica PIN pad, including all ten digits (i.e., all the digits must be included in the training set). Without losing generality, the attacker can use a USB PIN pad that logs the keys pressed and the corresponding timestamps. The attacker uses this information to segment the videos and labels them. Leveraging the logs, the attacker builds a training set containing, for each key pressed, a sequence of frames and the corresponding label (digits). Finally, the attacker trains the predictive model on the collected training set. For a detailed discussion on the implemented model, we refer readers to Section 4.5.

Phase B – Video Recording

The attacker hides a camera near the target ATM to record the PIN pad. There are multiple places where the camera can be placed, and depending on this, the attack can be easier or more challenging to succeed. The camera records the victim while entering the PIN and covering the PIN pad with the non-typing hand. The attacker retrieves the recorded video from the remote camera.

Phase C – PIN Inference

The attacker's goal is to infer the victim's PIN based on the video recorded during the PIN entering. First, the attacker

retrieves the timestamps from the recorded video. The attacker can use both the pressed keys' feedback sound or the masking symbols appearing on the screen while the victim enters the PIN to perform this task. Leveraging the timestamps, the attacker performs the same procedure as in Phase A to generate an attack set. Differing from the training set, the attack set contains a sequence of frames for each victim key pressed but no information about the related label. The adversary detects in the attack set the frames corresponding to a PIN entry, and splits the video into N sub-sequences where N represents the number of digits composing the PIN. For each sub-sequence, the adversary applies the model trained in Phase A. The model provides the probability of each class (i.e., the ten possible digits) to be the one corresponding to the input sub-sequence. Exploiting the N sub-sequences predictions, the attacker builds a rank of PINs in the descending order of their probabilities. In particular, the probability of a PIN corresponds to the product of the predicted probabilities of its digits.

3.2 Attack Settings

We consider three realistic attack scenarios:

1. **Single PIN pad** scenario: the attacker knows the model of the target PIN pad and obtains a copy of it to carry out the training phase. While this scenario may seem unrealistic, we note it is not difficult to obtain a specific keypad copy. Indeed, the attacker can easily obtain information about the ATM to be attacked and then buy the keypad with the same layout. Naturally, there can be certain differences concerning how sensitive the keypad is (for instance, due to usage, pads can become somewhat more difficult to press), but our experiments indicate such differences are not substantial enough to pose issues for deep learning models.
2. **PIN pad independent** scenario: this is the most challenging scenario. The attacker does not know or cannot retrieve the model of the target PIN pad. The training phase is performed on a PIN pad with similar characteristics to the target (e.g., shape, distance between keys, keys layout, and the sensitivity of keys).
3. **Mixed** scenario: as for the *Single PIN* scenario, the attacker knows the target PIN pad model. In this case, the training is performed on two PIN pads: a copy of the target and at least one PIN pad with similar characteristics. Considering several keypads in the training set makes sense when 1) the attacker is not certain about the keypad model, 2) the attacker assumes that the keypad will behave differently due to environmental conditions, 3) the attacker aims to attack multiple types of keypads (ATMs) with the same machine learning model, and 4) for any reason, the attacker did not manage to obtain enough training examples with a single keypad. We also note that using more keypads in the training set makes the training process more difficult and reduces the chances to overfit (i.e., we can consider different keypads as one keypad with noise, having the regularization effect [6]).

3.3 Camera Positions

Since our threat model allows the arbitrary position of the camera, we discuss several representative scenarios. We consider positions at the top of the ATM preferable for the attacker as lower positions of the camera result in no visibility of the hand pressing the keys if the other hand is covering it. We also consider settings at the front side of the chassis as they give better visibility for the attacker and are significantly more difficult for the victim to notice the camera.

Then, without loss of generality, we can discuss three main positions for the camera to provide good results. The camera can be positioned in the top left, center, or right corner. If the camera is positioned in the right corner and the person entering the PIN is right-handed, it will be easier to observe the entered digits. The same happens for the camera in the left corner and the left-handed person. However, if the camera is in the center position, it does not favor any specific setting, making it the most general setting, but it also makes it somewhat more challenging to conduct the attack than the left/right position and left/right-handed persons. We will concentrate on the top center position of the camera mounted on the chassis's front side.

4 Experimental Setting

To assess the feasibility of our attack on all the scenarios described in Section 3, we collected two datasets containing videos of people covering their typing hands while entering PINs. This section first illustrates the differences between the considered PIN pads and then describes our data collection procedure. Finally, the adopted video pre-processing, the setup used to run the experiments, and the implemented deep learning models are presented.

4.1 Devices under Test

We performed two separated data collection campaigns on two different real-world ATM metal PIN pads: DAVO LIN Model *D-8201 F* [5] (Figure 2a) and Model *D-8203 B* [6] (Figure 2b). In particular, we report the following differences between the two PIN pads:

- Model *D-8201 F* has a dimension of 100 mm x 100 mm, while Model *D-8203 B* has a metal surface of 92 mm x 88 mm and is contoured by rubber protection.

[5] https://www.davochina.com/4x4-ip65-waterproof-industrial-metal-keypad-stainless-steel-keyboard-for-access-control-atm-terminal-vending-machine-p00103p1.html

[6] https://www.davochina.com/4x4-ip65-stainless-steel-numeric-metal-keypad-with-waterproof-silicone-cover-p00126p1.html

- The horizontal key spacing is 1 mm larger between each key in Model *D-8203 B*.
- The keys of Model *D-8203 B* are harder to press and slightly taller than Model *D-8201 F*.
- For usability reasons, both the PIN pads emit a specific feedback sound (the same for all keys) when a key is pressed. The frequencies of the feedback sounds are 2 900 Hz for Model *D-8201 F* and 2 500 Hz for Model *D-8203 B*.

For the data collection, we embedded the PIN pad into a simulated ATM (see Figure 3). We chose the simulated ATM's size based on a real-world ATM [16]. In particular, the simulated ATM has a width of 60 cm, a height of 64 cm, and a depth of 40 cm. At 15 cm of height from the frame's base, we inserted a shelf to position the PIN pad and the monitor. The height of the PIN pad from the ground is 110 cm. We used three *Logitech HD C922 Pro* webcams anchored on the ATM's chassis to perform the video recording. A central webcam was placed 30 cm above the PIN pad, while the other two webcams were placed on the two top corners of the chassis 42 cm away from the PIN pad. The camera's maximum resolution is 1 080p with an acquisition rate of 30 fps. We recorded the videos with a resolution of 720p and an acquisition rate of 30 fps.

(a) *DAVO LIN Model D-8201 F* (b) *DAVO LIN Model D-8203 B*

Figure 2: The PIN pads used in the data collection.

4.2 Data Collection

The first data collection involved 40 participants (age 38.23 ± 11.43, 24 male and 16 female). The second data collection involved 18 participants (age 29.50 ± 5.74, ten male and eight female). Both collections include right-hand participants only. All the participants gave their approval to collect and use the data by signing informed consent. All the data have been anonymized and used by the authors of this paper for research purposes only. Participants were asked to stand in front of the test ATM and cover the typing hand while entering the PIN during the experiment. The participants were left free to type as they pleased. The goal is to emulate an ATM user that is hiding the PIN, preventing possible shoulder-surfing attacks. Each participant typed 100 5-digits PINs randomly generated, divided into four sequences of 25 PINs. This split into four sequences has been performed to include short breaks in the experiments and prevent the participants from getting tired. The PINs were showed one at a time on the ATM screen: once a PIN has been entered on the PIN pad, the user had to press the enter button to move to the next PIN. We recorded a total of 5 800 random 5-digit PINs, resulting in a balanced dataset per digit. Since our study aims to reconstruct the PIN from the video sequence, regardless of the user's typing behavior and familiarity with the PIN or the PIN pad, we decided to randomize PINs rather than asking users to enter the same PIN multiple times. This approach generalizes the attack, which can be applied to mnemonic PINs and One-time Passwords (OTPs). Moreover, we collected the environmental audio (exploiting the webcam microphone) and the keylogs of the PIN pad through the USB interface during the experiment. In particular, for each digit entered, we collect both the key down and key up events. We synchronized the video recordings with the timestamp of the key events. This information was collected to build the ground truth for the conducted experiments. The dataset is available at https://spritz.math.unipd.it/projects/HandMeYourPIN.

Figure 3: Our experimental setup. The cameras are visible but they can be hidden into the frame of an ATM. In all other aspects, we reproduced a common ATM layout in detail.

4.3 Pre-processing Video

Once the data acquisition phase is done, we need to pre-process the videos. For each video frame, we applied the following steps: i) convert the video frames to grayscale; ii) normalize the input so that all pixel values lie in the range $[0, 1]$; iii) crop the frames by centering the PIN pad, cutting off the irrelevant part of the background; (iv) resize the image to 250 x 250 pixels. After these steps, we applied a segmentation on each PIN video to obtain sub-sequences of frames corresponding to a single keypress (e.g., 5 sub-sequences for a 5-digit PIN). We extracted the keypress's timestamp from the recorded feedback sound of the PIN pad following the

procedure explained in [8]. In particular, we filtered the audio signal using a band-pass filter, centered on the specific frequency of the feedback sound (i.e., 2 900 Hz for Model *D-8201 F* and 2 500 Hz for Model *D-8203 B*). By identifying the peaks of the filtered signal, we could detect the timestamp of the target key (TK). This allowed us to extract a set of frames in each TK neighborhood. For each TK, the maximum number of frames (full-neighborhood) consists of all the frames ranging from the key preceding the TK to the key following the TK. If the TK corresponds to the first digit of the PIN, we consider only the frames between the TK and the next keypress. Analogously, if the TK corresponds to the last digit of the PIN, the frames considered are only those between the TK and its previous keypress. Since our model requires all input samples to have the same length, we decided to keep 11 frames for each sample. This value corresponds to the average number of frames in the full-neighborhood after removing the outliers over 3σ. To keep the TK at the center of the frames' sequence, we decided to consider five frames preceding the target keypress and five frames succeeding it, for a total of 11 frames per sample (including the target frame). There are three borderline cases: the TK is the first digit in the sequence, the TK is the last digit in the sequence, and the full-neighborhood has less than 11 frames. We apply black frame padding to keep the TK at the center of the sequence for these cases. In particular, if the TK is the first digit of the pin, five black frames are added at the head of the sequence, while if TK is the last digit of the PIN, we add five black frames at the end of the sequence. Finally, if there are not 11 frames in a sequence, we pad both the head and the tail (so that the TK is at the center).

4.4 Machine Learning Setup

For our experiments, we used a machine equipped with a CPU Intel(R) Xeon(R) E5-2670 2.60GHz, 128GB of RAM, and three Tesla K20m where each GPU has 5 Gb of RAM. To implement the machine learning models, we used Keras 2.3.0-tf (Tensorflow 2.2.0) and Python 3.8.6.

4.5 Prediction Models

Our approach aims to predict which key has been pressed on a PIN pad, exploiting only the video of a user covering the typing hand with the other hand. Since we deal with sequences of images, we implemented a model using Convolutional Neural Networks (CNNs) [24] and a Long Short-Term Memory (LSTM) [15]. The CNNs perform spatial feature extraction for each frame of a sequence, while the LSTM exploits these features to extract temporal patterns for the whole sequence of frames. The output of the LSTM passes through a multilayer perceptron (MLP) and a final Softmax activation function layer with ten units (as there are ten digits). This model is known in the literature as Long-term Recurrent Convolutional Network (LRCN) [11]. In Keras [20], such architecture can be implemented using the TimeDistributed wrapper throughout all the CNNs layers, which causes the same convolutional filters to be applied to all the timesteps (i.e., the frames) of the input sequence.

We split our dataset into train, validation, and test sets. Each set's size depends on the attack scenario and is discussed in detail in Section 5. We explored different hyperparameters by using the randomized grid search. Based on a preliminary assessment, we set the ranges for specific hyperparameters (i.e., we limit the upper value for specific hyperparameters) to speed up the search. In particular, for the CNNs, we tested $[3\text{x}3, 6\text{x}6, 9\text{x}9]$ kernel sizes. We also varied the number of convolutional layers in the range [1, ..., 4]. In the following dropout layer, we varied the dropout rates in the range $[0.01, 0.05, 0.1, 0.2]$. For the LSTM architecture, we varied the number of layers in the range [1, ..., 3], and the unit size in $[32, 64, 128, 256]$. We also assessed our network's performance using a Gated recurrent unit (GRU) instead of the LSTM. Finally, we examined the number of layers for the MLP in the range 1 to 4 and the number of units in the range $16, 32, 64, 128$. We tried two types of architectures for MLP: i) all the layers have the same number of units, ii) layers with decreasing number of units (funnel architecture), with every next layer having half the units of the previous one.

After a tuning phase, we selected a structure consisting of four convolutional layers (Conv2D in Keras) with ReLU activation functions, each followed by a pooling layer (MaxPooling2D in Keras). Three convolutional layers have a filter size of 3x3, and one (the second one) has a filter size of 9x9. Each pooling layer has a filter size of 2x2. The number of filters in the convolutional layers doubles at each layer, starting from 32 filters in the first layer, ending up at 256 filters in the fourth layer. We added a dropout layer (dropout rate 0.1) after the last pooling layer to prevent overfitting. The output is then flattened, preserving the temporal dimension to provide a sequence of temporal features to the following LSTM. A single layer LSTM with 128 units resulted in the best validation with a hyperbolic tangent activation function. Finally, for the MLP, we used four fully connected layers, with 64 units each, followed by the Softmax activation layer with ten units (i.e., the number of classes we want to predict). We used the categorical cross-entropy loss function and the stochastic gradient descent (SGD) optimizer. Finally, we set the model to evaluate the accuracy metric. We set the batch size to 16 and the learning rate to 0.1. We tested for 70 epochs since we found that the model always converged within this number of epochs. In Appendix A, we provide additional details. Our experiments indicate that the classification task we conduct is relatively difficult, and one needs to use sophisticated deep learning architecture for good results. Still, we note that the architecture we use is in line with the state-of-the-art results for hand tracking problem [17, 23]. Finally, we observed significant changes in the performance depending on the specific

hyperparameter choice, indicating a need for detailed tuning for the respective tasks.

In a real-world context, it might not be possible to reproduce precisely the experimental conditions (e.g., the camera might be rotated/tilted slightly concerning the PIN pad, or the distance to the PIN pad might not be the same). Thus, we also used data augmentation to generate synthetic measurements (20% of the training dataset) that cover more scenarios to account for such issues. In particular, we used the following video-based transformations:

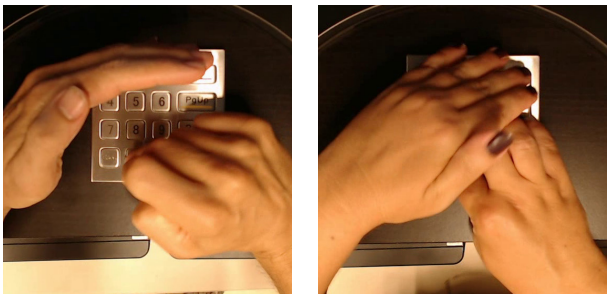
- **rotation** for a maximum of 7 deg both clockwise and counterclockwise;
- **horizontal shift** for a maximum of 10% of the width;
- **vertical shift** for a maximum of 10% of the height;
- **zoom** between 0.9 and 1.1.

Synthetic samples were generated by randomly combining the transformation techniques listed above. We emphasize that data augmentation is also helpful as it makes the predictive model adaptable to different types of ATMs.

5 Experimental Results

In this section, we evaluate the performance of our approach for the three attack scenarios described in Section 2. We adopted a user-independent split strategy since, in a realistic context, the attacker does not have labeled videos of victims entering PINs. In this way, we guarantee that videos from a participant appear only once among the three sets. Moreover, since we are interested in evaluating the PINs reconstruction accuracy, we removed all non-5-digit sequences entered by mistake by participants (i.e., the "enter" key was pressed after a sequence longer or shorter than 5-digits.) The removed non-5-digits sequences account for 2.2% of the total PINs entered. We conducted the experiments on both 4-digits and 5-digits PINs. To experiment on 4-digit PINs, we removed the last digit of each 5-digit sequence in our dataset.

We define that a PIN is covered when there is no direct view of the entered keys and their surrounding. Still, we observed that some participants failed to obtain a satisfactory coverage level with the non-typing hand despite our instruction before starting the data collection. Since this study aims to infer covered PINs, we decided to exclude the videos of participants that entered badly covered PINs from the validation and test sets. In this way, the validation and test sets consist of videos of covered entered PINs, while the training is composed of videos containing both covered and badly covered PINs. Note that badly covered PINs are still difficult to "read" by simply looking at the video, so we consider such data useful in building a training set. For the test set, we aim for the most difficult scenario where PINs are properly covered. Under these assumptions, we "blacklisted" 16 participants that badly covered the PIN pad: 14 for the first data collection and two for the second data collection. These participants have been excluded from validation and test sets described in the below



(a) *Badly covered PIN that we excluded from the validation and test tests.*

(b) *Covered PIN, where there is no direct view of the pressed key and the surrounding digits.*

Figure 4: Badly covered vs. covered PINs.

scenarios [7]. In Figure 4, we provide an example of a badly covered PIN and a covered PIN.

To obtain a further indication of the quality of coverage and the difficulty of reconstructing a PIN by a human, we surveyed a random sub-sample of videos of covered PINs (Section 7). Finally, there is a question of how to predict the PIN that is not guessed correctly from the first attempt. Since we consider each digit independently, we consider a mechanism where our best guess comprises of individual best guesses (for each digit). If that PIN is incorrect, we consider the digit where the two best guesses have the smallest difference. We change that digit to the second-best guess in our PIN, and we try again. The same procedure is repeated for the third attempt if the second PIN is wrong.

1. **Single PIN pad scenario**. To evaluate the scenario where the adversary knows the target PIN pad model and owns a copy, we considered only the first data collection composed of 40 participants. We applied a user-independent split of the dataset in training, validation, and test sets with the proportions 80/10/10%.
2. **PIN pad independent scenario.** In this scenario, the adversary trains the machine learning model on a PIN pad with a similar layout to the target one. This scenario occurs when the attacker cannot obtain the same PIN pad model to collect data. Under these assumptions, we used for training and validation the first collected dataset (composed of 40 participants). We included the videos from 35 participants in the training set and the remaining 5 participants' videos in the validation set. We used the second collected dataset as the test set. We included only the videos of 16 out of 18 participants of the second data collection in the test set since two were in the group that badly covered the PIN pad.
3. **Mixed scenario.** This scenario corresponds to how the attacker owns both a copy of the target PIN pad and a PIN pad similar to the target one. In this case, we

[7] Results are in Appendix C

(a) *True digit = 7*
Pred = 7 (0.999), 4 (0.000), 8 (0.000)

(b) *True digit = 3*
Pred = 3 (0.979), 2 (0.012), 6 (0.005)

(c) *True digit = 6*
Pred = 6 (0.819), 9 (0.170), 8 (0.009)

(d) *True digit = 3*
Pred = 3 (0.809), 2 (0.092), 5 (0.069)

(e) *True digit = 3*
Pred = 2 (0.329), 3 (0.315), 6 (0.185)

Figure 5: PIN 73633 entered by a user in our test set in the *Single PIN pad* scenario. Our algorithm suggests 73632 as the most probable PIN (probability = 21.32%), 73633 as the second most probable PIN (probability = 20.43%), and 73636 as the third most probable PIN (probability = 11.96%). The algorithm predicts the correct PIN in the second attempt.

merged the two collected datasets and applied a user-independent split in training, validation, and test sets with the proportions 80/10/10%.

We begin the discussion on results by providing an example of a successful PIN attack in Figure 5. We consider the 5-digit PIN case and the *Single PIN pad* scenario. We provide an image for each digit. We give the top three digits and the corresponding accuracy values. Notice how the first and second digits are predicted correctly with high probabilities. This happens as the person sets the hand to allow an easy start of typing. Already for the third digit, we observe a significant drop in the accuracy value for the best prediction. Still, the value is significantly larger than the second-best prediction, so there are no issues in getting the correct prediction. This trend continues for the fourth digit and gets very pronounced for the last (fifth) digit. Indeed, the best guess is not correct anymore, but the second-best guess is correct (the difference in probability between those two guesses equals 0.014).

For all three scenarios, Figure 6 shows the results for the single key accuracy, while Figure 7 reports the results considering 5-digit and 4-digit PINs. Considering the single key accuracy (averaged over all digits), notice that even in the most difficult *PIN pad independent* scenario, our Top-3 accuracy reaches 63.8%, which is significantly higher than the result one would reach with random guessing (30%). At the same time, the results for the *Single PIN pad* scenario and the *Mixed* scenario are rather similar, and the Top-3 accuracy reaches up to 88.7%. Interestingly, we observe somewhat better results for Top-2 and Top-3 accuracy for *Single PIN pad* scenario than the *Mixed* scenario, which is the opposite of the results for 4-digit and 5-digit settings. We hypothesize this happens as we consider independent digits as naturally, the best results happen when the training and test are done on the same device. On the other hand, the *Mixed* scenario gives slightly better results for the PIN reconstruction scenarios as we need to consider a sequence of PINs with the movement between digits. Then, having different devices in the training set allows (slightly) better generalization.

In Figure 7a, we observe that the most difficult case is when the attacker does not have access to the same keypad as used by the victim. There, the accuracy for the Top-3 case equals 11.4%. Having access to the same type of keypad improves accuracy in Top-3 to more than 20%. Finally, considering the *Mixed* scenario, we can improve the accuracy for Top-3 to almost 30% (29.7%). Next, in Figure 7b, we present results for 4-digit PINs. The results are significantly better than for the 5-digit scenario. The lowest accuracy happens for the Top-1 *PIN pad independent* scenario setting and it equals 10.6% (cf. 6.7% for the 5-digit scenario). The highest accuracy reaches 41.1% for the Top-3 accuracy in the *Mixed* scenario.

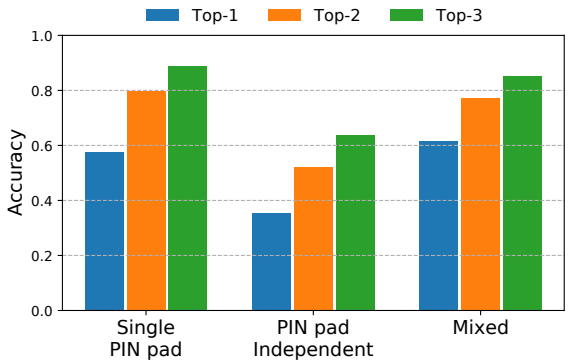
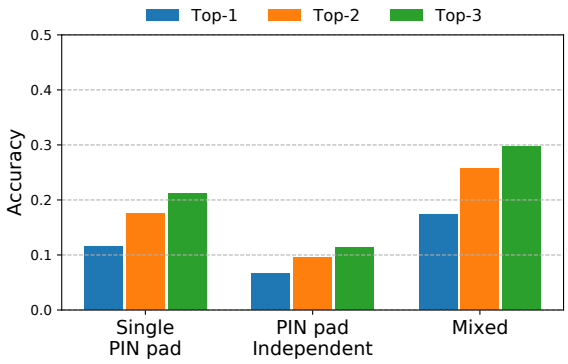
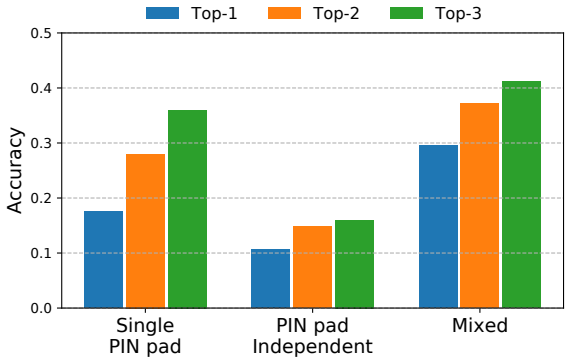


Figure 6: Single key accuracy of our algorithm for the three considered attack scenarios. Top-N means that we guessed the digit within the *N* attempts.

In Figure 8, we depict detailed results for the digit 1. We selected this digit since heat maps for others look similar and exhibit similar dispersion. First, in Figure 8a, we show the PIN pad layout. Figure 8b gives results for the *Single PIN pad* scenario. Notice that the heat map indicates that guess 1 is the most likely one with 67% probability. The digits 4 and 3 are recognized as the second and third best guess, respectively.



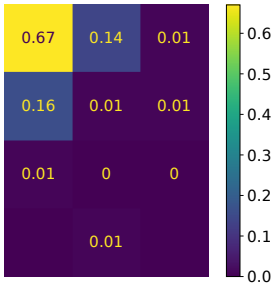
(a) *5-digit PINs.*



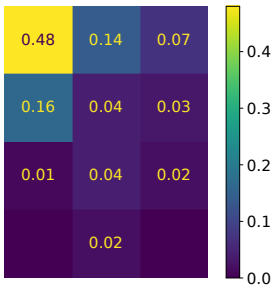
(b) *4-digit PINs.*

Figure 7: PIN accuracy of our algorithm in the three considered attack scenarios. Top-N means that we guessed the PIN within the N attempts.

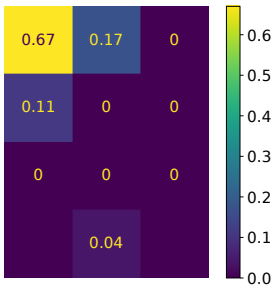
(a) *Layout of a generic PIN pad.*



(b) *Single PIN pad scenario.*



(c) *PIN pad independent scenario.*



(d) *Mixed scenario.*

Figure 8: Digit 1 predictions heat maps for the three considered attack scenarios.

Still, their probability is significantly lower. For the *PIN pad independent* scenario, we observe that the probabilities are more spread over all digits, which comes at the expense of a lower prediction probability for the correct digit. The second and third best guesses maintain the probabilities, indicating that Top-3 guesses are sufficient to guess a large number of PINs in the most difficult scenario. Finally, Figure 8d gives results for the *Mixed* scenario, where we see that the best guess is on the level with the *Single PIN pad* scenario. Interestingly, now the second and third best guesses are swapped compared to the previous scenarios. All the other digits have 0 or negligible probability of being the correct digit. Appendix B provides additional results for the key accuracy.

Based on our results, we provide several observations that we believe generalize beyond these experiments:

- Covering the PIN pad with the other hand is not sufficient to defend against deep learning-based attacks.
- Portability aspect (keypad differences) is quite significant, and the attacker should obtain the same type of keypad for a high probability of success in attack.
- There are three prevailing ways how users cover the typing hand: raised hand not touching the surface, hand resting on fingers and vertically covering the PIN pad, and hand resting on the side of the palm. The examples of all three covering strategies are shown in Figure 9.

Finally, Table 1 provides a comparison between our attack and several unobtrusive attacks on 4-digit PINs from the literature [8]. We divided the attacks according to the information that the attacker has: keystroke timing (KT), one digit of the victim's PIN (OD), and the thermal trace (TT) left on the PIN pad by the victim [1]. From the results, it is clear that our attack performs the best for all considered TOP-N accuracies.

Appendix C provides experiments where we: i) resize the images, ii) consider different camera positions, iii) consider setup without data augmentation, and iv) consider the training set that includes the blacklisted participants. Finally, we also provide experiments for the frame detection error (when the feedback sound is not properly synchronized).

6 Countermeasures

Different countermeasures could make the attack more difficult to succeed. For instance:

1. Longer PINs. This countermeasure would make the attack more difficult, as evident from the comparison for 4- and 5-digits PINs. This countermeasure would be relatively easy to support from a technical perspective. At the same time, it would have usability drawbacks as longer PINs take more time to type and are more difficult

(a) *Side: hand resting on the side of the palm.*

(b) *Over: raised hand not touching the surface.*

(c) *Top: hand resting on fingers and vertically covering the PIN pad.*

Figure 9: Different covering strategies using the non-typing hand.

Attacker Information Source				4-digit PINs TOP-N Accuracy (%)		
KT	**OD**	**TT**	**Our Attack**	**TOP-1**	**TOP-2**	**TOP-3**
				0.01	0.02	0.03
	§			0.10	0.20	0.30
§				0.02	0.35	0.72
§	§			3.02	3.72	4.36
		§		3.76	7.52	11.28
§		§		15.54	27.79	33.63
			§	**29.61**	**37.06**	**41.12**

Table 1: Comparison of our attack with other unobtrusive attacks on ATM PIN pads. Note that we need to extract the frame for our attack, while for KT, one needs to use the timestamp, which is more precise information.

to remember.

2. Virtual and randomized keypad. Instead of using a mechanical keypad, one could consider using a touchscreen where the digits are randomized. More and more ATMs (but not PoS) have this feature, so implementing it would not be too difficult. Unfortunately, we believe this would seriously damage the usability aspect as people are accustomed to digits occurring in the natural sequence, and any changes would probably result in wrongly entered PINs.
3. Screen protectors. On many ATMs, there are already various types of screen protectors that occlude the typing hand. To maintain usability, many screen protectors are short and will not cover the whole typing hand. Making the screen protectors larger would impair usability as it will become more difficult for the user to read the keypad. This countermeasure is potentially not easy to deploy as it requires physical changes to the ATMs.

Next, we analyze how a PIN shield could affect the performance of our attack. We simulated the presence of the shield by applying a black patch to cover the PIN pad. In Table 2 we report the performance of our attack in the *Mixed* scenario, applying four different levels of coverage (Figure 16, Appendix C). The coverage of the PIN pad is larger than the percentage shown in Figure 16 since the coverage given by the non-typing hand is not included in the given percentage. The results show that our attack remains effective even when 75% of the PIN pad is covered, while the performance decays significantly beyond this level of coverage. As such, it becomes clear that our deep learning attack uses information about the whole hand position and movement, and not only the tip of the fingers. Since the last row of the PIN pad has only one number (0), 100% coverage has poor attack results not only because of hiding all the numbers on the keypad but due to hiding of proximal interphalangeal, metacarpophalangeal, and carpometacarpal joints of the fingers. Thus, only PIN shields that offer full PIN pad coverage can be considered effective countermeasures to our attack.

Coverage percentage	**Key accuracy**	**PIN TOP-3 accuracy**
25%	0.54	0.22
50%	0.55	0.22
75%	0.50	0.17
100%	0.33	0.01

Table 2: PIN shield experiments.

We provide additional results with different covering strategies (`Side`, `Over`, and `Top`) in Appendix C. Those results again show that covering the PIN pad from the `Side` gives insufficient protection. On the other hand, using the `Over` strategy significantly decreases the key accuracy and PIN accuracy.

7 Deep Learning vs. Humans

If an attacker has direct visibility of the PIN pad, reconstructing a PIN from a video can be considered a trivial task. One of

the classic countermeasures to the so-called shoulder-surfing attacks is to cover the hand entering the PIN with the non-typing hand. In this way, the victim obstructs the attacker by removing the direct visibility of the keypad. We designed a questionnaire to evaluate how much the covering with the non-typing hand effectively prevents the PIN reconstruction.

7.1 Methodology

The questionnaire consists of 30 videos of people entering 5-digit PINs by covering the PIN pad with the non-typing hand as we noticed that for longer questionnaires, the participants' attention significantly goes down toward the end. For each video, the participants had to indicate the three most likely PINs in their opinion.

To assess human and model performance on both the PIN pads, we decided to use the test set of the *Mixed* scenario (i.e., the only one including both PIN pads). Since the test set was balanced in terms of samples per user, we randomly selected five PINs for each of the six users in the test set. We extracted 30 videos corresponding to the selected PINs from our dataset. We kept the original resolution of 720p and the original audio track containing the feedback sound emitted by the PIN pad for each video. The feedback sound helps the participants to recognize when a digit is entered. To avoid bias in the answers, we randomized the order of the videos in the questionnaire. Moreover, the participants were free to modify all their answers until the final submission. We did not apply any particular restriction to the participants during the filling of the questionnaire. In particular, there were no time restrictions to complete the task. The participants could freely apply the strategy they prefer to infer the PIN (e.g., write down the digits, pausing the video, restart the video any number of times, use the slow-motion option). Finally, we provided the users with the layout of the PIN pad.

To evaluate if people with specific knowledge about the task achieve a better performance, we pre-trained a group of participants. Specifically, we provided participants with a new set of 20 videos of users typing PINs by covering the PIN pad with the non-typing hand and the corresponding typed PIN. To make the training more effective, we decided to provide participants with videos of users included in the questionnaire (none of the videos are present in both training and questionnaire). Additionally, the questionnaire had suggestions on what to pay special attention. For a participant to be considered trained, the complete viewing of all 20 videos is required. In addition, trained participants could also watch the training videos while filling the questionnaire.

7.2 Evaluation and Discussion

A total of 78 distinct participants took part in our questionnaire experiment. In particular, 45 participants (14 female age 34.1 ± 10.4 years and 31 male age 29.7 ± 8.3 years) completed the experiment without any training, while 33 participants (10 female age 29.1 ± 3.3 and 23 male age 29.3 ± 5.6) completed the experiment after the training session. None of the questionnaire participants took part in the two data collections described in Section 4.2.

The proposed questionnaire's goal is twofold: i) investigate how effective the hand coverage is in preventing a PIN from being inferred by a human, and ii) compare the performance of our deep learning approach with that of a human. Although the coverage of the PIN pad provides an obstacle to the immediate identification of the typed PIN, a human can exploit various information (both local and global) to reduce the probability space about where to look for the entered PIN:

- Knowing the keys' spatial positioning thanks to the given layout of the target PIN pad.
- Understanding which finger pressed the key from the movements of the hand.
- Evaluating the topological distance between two consecutive keys from the feedback sound emitted by the PIN pad. Specifically, two topologically close keys have temporally close sound feedback [8].
- Excluding keys based on the non-typing hand coverage.
- Guessing the finger position based on the hand displacement between the insertion of a key and the next one.
- Deducing the fingers' position of the covered hand.

Although a human can exploit this information, the PIN pad coverage still partially prevents PIN reconstruction. In particular, the participants in our questionnaire could reconstruct on average (of both trained and non-trained humans) only 4.49% of the PINs entered in the videos on the first attempt and 7.92% within three attempts. The performance increasing between Top-1 and Top-3 accuracy suggests a certain ability in estimating the neighborhood of the keys pressed. This ability is also highlighted in Figure 11a, where the probability distribution shows how the error decreases with the increase of the topological distance from the target key. The heat maps for other keys look similar and exhibit similar dispersion.

Unlike humans, our algorithm focuses on target key classification and then reconstructs the entire PIN sequence. To compare the model's performance to that of humans on the same task, we evaluated our algorithm's accuracy on the videos included in the questionnaire. Recall that the questionnaire's videos are a sub-sample of the *Mixed* scenario test set, and therefore were not used in the model training phase. As reported in Figure 10, our model performs better than humans in all Top-N accuracy scenarios. To evaluate if our algorithm performance and humans' performance in reconstructing 5-digit PINs are statistically different, we applied a series of Chi-square tests [27]. The Chi-square test resulted significant for all Top-1 ($\chi^2 = 14.19, p < 0.001$), Top-2 ($\chi^2 = 15.84, p < 0.001$), and Top-3 ($\chi^2 = 21.37, p < 0.001$) accuracy values for non-trained humans. In particular, our model outperforms humans showing a four-fold improvement in reconstructing a PIN in three attempts. Similarly, for trained

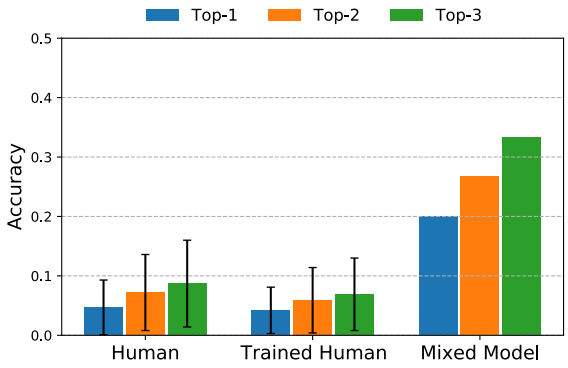
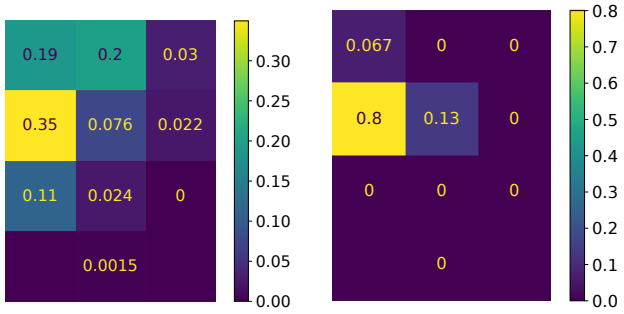


Figure 10: Comparison between human (non-trained and trained) and deep learning model performance in the subset of videos included in the questionnaire. Top-N means that participants guessed the PIN within the N attempts.



(a) *Humans.* (b) *Mixed* scenario model.

Figure 11: Digit 4 predictions heat maps for the videos included in the questionnaire. We report an example from non-trained humans, since the heat maps for both non-trained and trained human are similar.

humans, the Chi-square test resulted significant for all Top-1 ($\chi^2 = 16.12, p < 0.001$), Top-2 ($\chi^2 = 20.83, p < 0.001$), and Top-3 ($\chi^2 = 28.88, p < 0.001$) accuracy values.

This result comes from the difference in performance in the classification of single keys. The human average accuracy (considering both human data collections) on single key classification equals 0.351, approximately half compared to the model key accuracy of 0.687. The comparison of Figures 11b and 11a shows how the error in identifying a digit is significantly higher for humans, justifying why the increase in Top-2 and Top-3 PIN accuracy is greater for our algorithm. Finally, comparing trained and non-trained humans, the Chi-square test reported no significant differences with $p > 0.1$ for all Top-1, Top-2, and Top-3 accuracy values. This means that training does not improve a human's ability to identify a PIN within three attempts. Potentially, either a longer training could be required, or additional feedback from an expert should be provided to improve the performance. Appendix B provides additional results for the comparison between our deep learning model and human performance.

8 Related work

Side-channel attacks specifically target the information gained by the implementation of a system [26]. Most of the time, these attacks exploit channels like sound [13], timing [22], power consumption [21], and electromagnetic emanations [4] to learn the system's secrets in use. In [22], the authors managed to crack RSA keys by carefully timing the operations performed by the key-generating algorithm. Another example of a timing attack is reported in [32], where the authors measured the timing between keystrokes in interactive SSH sessions in an attempt to retrieve the typed passwords.

Human behavior can also be defined as a side-channel of a system, especially if the analyzed behavior directly results from the system's requirements. In [3], the authors analyzed the hand movements of people typing on a keyboard and, by using basic computer vision techniques, they tried to reconstruct the text being typed. In [31], the authors again analyzed the finger motion during the PIN-entry process on smartphones. They showed that 50% of the 4-digit PINs could be retrieved in just one attempt. Different from our work, where the target of the attack is a physical PIN pad, in [31], the attackers could also exploit more information. In particular, the users typed the PIN using only one finger, and the attacker knew the finger the users are typing. The different contexts and assumptions make the works substantially different. In [33], the authors presented a side-channel attack on tablets, consisting of analyzing the backside movements of the tablet itself to infer what is being typed by the victim. To do so, they selected some peculiar features of the backside of the tablets (e.g., logos, side-buttons) and analyzed their movement throughout the frames to understand what area of the virtual keyboard is being pressed. Similarly, in [34], the authors presented an attack to infer the pattern lock of mobile devices from videos. Different from our approach, in [34], the attacker required a vision of the user's fingertip while drawing the pattern and a part of the device.

PIN and PIN pad attacks represent a branch of side-channel attacks that exploit information leakage from keyboards and numeric keyboards (i.e., PIN pads) to infer what the victim has typed (e.g., passwords or PINs). In this context, some works focused on exploiting the heat transferred from the hand to the keypad when the victim enters the PIN or password [19, 28]. The attacker points a thermal camera to the keypad as soon as the victim has finished entering the PIN. The thermal image shows which keys have been pressed and even highlights the order in which the victim pressed them. The main advantage of this attack is that it does not require the attacker to do anything while the victim is typing the PIN.

On the other hand, the attacker must act quickly (i.e., within seconds) for a higher success rate as the heat on the keypad rapidly fades away. Another drawback of the attack is that its effectiveness depends on the keypad's material (e.g., metal PIN pads completely nullify the attack because of their high thermal conductivity).

Timing attacks against PINs represent another type of side-channel attack against this authentication method. In the scenario presented in [2], the attacker recorded the screen of an ATM while the victim is entering the PIN. When analyzing the recorded video, the attacker exploited the PIN masking symbols appearing on the ATM screen to extract timing information about the keystrokes. The attacker used predictive models to infer which keys were most likely typed by the victim, starting from the deduced inter-keystroke timing. In [8], the authors used the ATM's sound whenever a button is pressed. ATM's sound must be independent of which button is being pressed (i.e., a generic feedback sound). This consideration means that one feedback sound will not help the attacker. However, the sound gives enough information to extract a timestamp of the keys being pressed. Moreover, in [8,25], the authors showed how combining timing, acoustic, and thermal information can significantly reduce the number of attempts to guess a PIN (e.g., 34% of 4-digits PINs are recovered in three attempts). These attacks need to be reevaluated from a feasibility perspective in a real-world setting. In particular, as shown in [19], the heat signature is dissipated abruptly by metal PIN pads. The lack of this information limits the performance of the attacks presented in [8,25], reducing the probability of guessing a 4-digit PIN in 3 attempts to 5%.

Our work shows several advantages over the state-of-the-art in ATM PIN inference. To the best of our knowledge, we are the first to investigate the security of hand covering protection methods for ATM's PIN entering. Further, our method shows a significant improvement in reconstructing the PIN compared to previous work on metal PIN pads, reaching 41% of success in reconstructing 4-digit PINs in three attempts (and correctly guessing every third PIN in the first guess).

9 Conclusion

This paper proposed a deep learning attack on PIN mechanisms reaching high accuracy even when the user covers the PIN to be entered. Our attack leverages the information from the hand position but also hand movements while entering the PIN. Our attack works in the profiling setup where the attacker uses a copy of the keypad to train the deep learning model and then attacks a different device while the victim is entering the PIN. For a 4-digit PIN, our attack reaches an accuracy of more than 40%, making it practically applicable and more powerful than the attacks from the related works.

Our data collection phase involved 58 persons, and our questionnaire involved 78 participants. While this required a significant effort and several months of data acquisition, one could still consider the datasets too small to allow general conclusions. Next, our analysis considered only two types of keypads. While most keypads do not have significant differences, including more keypad models in our analysis would be interesting. Additionally, there are several potential sources of bias in our data collection phase. While we managed to get a relatively good male and female participants ratio, we notice that data is skewed from several perspectives. Unfortunately, this was not possible to avoid as the participation was voluntary [8].

1. Our dataset has users ranging from 24 to 50 years. While this provides good variety, it would be good if it included older people. Still, we do not expect any difficulties in running our attack. We consider it even somewhat easier as we noticed older people make more significant hand position adjustments when entering the PIN.
2. Our analysis includes only right-handed persons. We do not expect any issues due to the dataset's limitations as we use a camera positioned in the center. Still, we expect the attack to be more difficult when attacking left-handed persons if the training set does not contain such examples. Finally, from the real-world practicality, there are approximately 90% of right-handed persons vs. 10% left-handed persons [29], so our attack generalizes for the dominant part of the population.
3. All participants were Caucasians. We expect our attack will have difficulties working for people from other races. Still, this can be alleviated by expanding the training set to include more racial diversity.

Possible future work includes:

1. In our data collection phase, we allowed the users to select their covering strategies. Based on the current results, it would be interesting to explore if modifications in how the user covers the PIN would allow more protection.
2. We noted several potential sources of bias in our data collection phase. Including participants from other races and left-handed persons would allow us to make more general conclusions.
3. To avoid the need that the attacker should have different keypads, it would be beneficial to assess whether some more straightforward solution like a paper copy of the keypad would suffice (at the expense of losing information about the keypress sensitivity).
4. It would be interesting to investigate if it is possible to extract the timestamp directly from the video (when a person clicks a button, there is a specific movement).

References

[1] Yomna Abdelrahman, Mohamed Khamis, Stefan Schneegass, and Florian Alt. Stay cool! understanding

[8] The 2021 COVID-19 situation made data acquisition more challenging as participants needed to be in our lab during the data acquisition.

thermal attacks on mobile-based user authentication. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 3751–3763, 2017.

[2] Kiran Balagani, Matteo Cardaioli, Mauro Conti, Paolo Gasti, Martin Georgiev, Tristan Gurtler, Daniele Lain, Charissa Miller, Kendall Molas, Nikita Samarin, et al. Pilot: Password and pin information leakage from obfuscated typing videos. *Journal of Computer Security*, 27(4):405–425, 2019.

[3] Davide Balzarotti, Marco Cova, and Giovanni Vigna. Clearshot: Eavesdropping on keyboard input from video. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 170–183. IEEE, 2008.

[4] Shivam Bhasin, Anupam Chattopadhyay, Annelie Heuser, Dirmanto Jap, Stjepan Picek, and Ritu Ranjan Shrivastwa. Mind the portability: A warriors guide through realistic profiled side-channel analysis. In *27th Annual Network and Distributed System Security Symposium, NDSS 2020, San Diego, California, USA, February 23-26, 2020*. The Internet Society, 2020.

[5] Farid Binbeshr, ML Mat Kiah, Lip Yee Por, and Aws Alaa Zaidan. A systematic review of pin-entry methods resistant to shoulder-surfing attacks. *computers & security*, page 102116, 2020.

[6] Chris M. Bishop. Training with noise is equivalent to tikhonov regularization. *Neural Comput.*, 7(1):108–116, January 1995.

[7] Mike Bond, Omar Choudary, Steven J Murdoch, Sergei Skorobogatov, and Ross Anderson. Chip and skim: cloning emv cards with the pre-play attack. In *2014 IEEE Symposium on Security and Privacy*, pages 49–64. IEEE, 2014.

[8] Matteo Cardaioli, Mauro Conti, Kiran Balagani, and Paolo Gasti. Your pin sounds good! augmentation of pin guessing strategies via audio leakage. In *European Symposium on Research in Computer Security*, pages 720–735. Springer, 2020.

[9] National Cash Register (NCR Corporation). The rise of emv and what it means for the magnetic stripe. https://www.ncr.com/blogs/payments/emv-magnetic-stripe, March 2021. [Online; accessed 7-June-2021].

[10] Belinda L Del Gaudio, Claudio Porzio, Gabriele Sampagnaro, and Vincenzo Verdoliva. How do mobile, internet and ict diffusion affect the banking industry? an empirical analysis. *European Management Journal*, 2020.

[11] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015.

[12] Malin Eiband, Mohamed Khamis, Emanuel Von Zezschwitz, Heinrich Hussmann, and Florian Alt. Understanding shoulder surfing in the wild: Stories from users and observers. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 4254–4265, 2017.

[13] Daniel Genkin, Adi Shamir, and Eran Tromer. Acoustic cryptanalysis. *J. Cryptol.*, 30(2):392–443, April 2017.

[14] Brij B Gupta and Shaifali Narayan. A survey on contactless smart cards and payment system: Technologies, policies, attacks and countermeasures. *Journal of Global Information Management (JGIM)*, 28(4):135–159, 2020.

[15] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[16] Nautilus Hyosung. cmax7600ta installation manual. http://www.tetralink.com/core/media/media.nl/id.46617/c.4970910/.f?h=d919934a85943438b8fe, 2015. [Online; accessed 30-December-2020].

[17] M. Islam, M. Hossain, R. ul Islam, and K. Andersson. Static hand gesture recognition using convolutional neural network with data augmentation. In *2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, pages 324–329, jun 2019.

[18] ISO. Iso: Financial services - personal identification number (pin) management and security - part 1: Basic principles and requirements for pins in card-based systems. https://www.iso.org/standard/68669.html, 2017. [Online; accessed 30-December-2020].

[19] Tyler Kaczmarek, Ercan Ozturk, and Gene Tsudik. Thermanator: Thermal residue-based post factum attacks on keyboard data entry. In *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*, pages 586–593, 2019.

[20] Nikhil Ketkar. Introduction to keras. In *Deep learning with Python*, pages 97–111. Springer, 2017.

[21] Jaehun Kim, Stjepan Picek, Annelie Heuser, Shivam Bhasin, and Alan Hanjalic. Make some noise. unleashing the power of convolutional neural networks for profiled side-channel analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 148–179, 2019.

[22] Paul C Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *Annual International Cryptology Conference*, pages 104–113. Springer, 1996.

[23] K. Lai and S. N. Yanushkevich. Cnn+rnn depth and skeleton based dynamic hand gesture recognition. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 3451–3456, 2018.

[24] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[25] Ximing Liu, Yingjiu Li, Robert H Deng, Bing Chang, and Shujun Li. When human cognitive modeling meets pins: User-independent inter-keystroke timing attacks. *Computers & Security*, 80:90–107, 2019.

[26] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer, December 2006. ISBN 0-387-30857-1, http://www.dpabook.org/.

[27] Mary L McHugh. The chi-square test of independence. *Biochemia medica*, 23(2):143–149, 2013.

[28] Keaton Mowery, Sarah Meiklejohn, and Stefan Savage. Heat of the moment: Characterizing the efficacy of thermal camera-based attacks. In *Proceedings of the 5th USENIX conference on Offensive technologies*, pages 6–6, 2011.

[29] Marietta Papadatou-Pastou, Eleni Ntolka, Judith Schmitz, Maryanne Martin, Marcus R Munafò, Sebastian Ocklenburg, and Silvia Paracchini. Human handedness: A meta-analysis. *Psychological Bulletin*, April 2020.

[30] Nolen Scaife, Christian Peeters, and Patrick Traynor. Fear the reaper: Characterization and fast detection of card skimmers. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pages 1–14, 2018.

[31] Diksha Shukla, Rajesh Kumar, Abdul Serwadda, and Vir V Phoha. Beware, your hands reveal your secrets! In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 904–917, 2014.

[32] Dawn Xiaodong Song, David A Wagner, and Xuqing Tian. Timing analysis of keystrokes and timing attacks on ssh. In *USENIX Security Symposium*, volume 2001, 2001.

[33] Jingchao Sun, Xiaocong Jin, Yimin Chen, Jinxue Zhang, Yanchao Zhang, and Rui Zhang. Visible: Video-assisted keystroke inference from tablet backside motion. In *NDSS*, 2016.

[34] Guixin Ye, Zhanyong Tang, Dingyi Fang, Xiaojiang Chen, Kwang In Kim, Ben Taylor, and Zheng Wang. Cracking android pattern lock in five attempts. In *Proceedings of the 2017 Network and Distributed System Security Symposium 2017 (NDSS 17)*. Internet Society, 2017.

A Neural Networks Additional Info

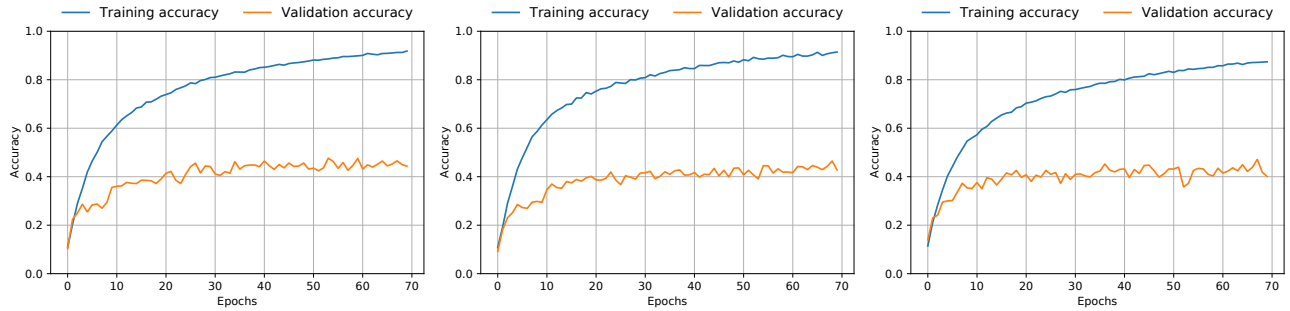
In Figure 12, we show the training and validation accuracy for the three models selected after the random grid search. In the *Mixed* scenario, the validation accuracy grows faster than in *PIN pad independent* scenario and *Single PIN pad* scenario, reaching faster the plateau. Indeed, in the *Mixed* scenario, the validation accuracy stabilizes after 20 epochs, while we require more than 35 epochs for the other scenarios. This difference can be linked to a larger training size and a higher variance in the samples since the *Mixed* scenario is the only one to include videos from both PIN pads in the training phase.

Next, we report some statistics about the training execution times for the three scenarios we consider.

- **Single PIN pad** scenario: the training set is composed of 32 participants, corresponding to 16 000 samples of 11 frames each. Our model takes 1 577 seconds to complete an epoch (i.e., approximately 34 hours to complete the entire training phase).
- **PIN pad independent** scenario: the training set is composed of 35 participants, corresponding to 17 500 samples of 11 frames each. Our model takes 1 598 seconds to complete an epoch (i.e., approximately 34 hours to complete the entire training phase).
- **Mixed** scenario: the training set is composed of 46 participants, corresponding to 23 000 samples of 11 frames each. Our model takes 2 240 seconds to complete an epoch (i.e., approximately 46 hours to complete the entire training phase).

B Key Accuracy Analysis

In this section, we provide further analysis on the key accuracy for our attack. Figure 13 highlights that the accuracy on a single key is worse in the *PIN pad independent* scenario. Although the performance is considerably lower than the



(a) *Single PIN pad scenario. We included 4 participants in validation, corresponding to 400 digits.*

(b) *PIN pad independent scenario, We included 5 participants in validation, corresponding to 500 digits.*

(c) *Mixed scenario. We included 6 participants in validation, corresponding to 600 digits.*

Figure 12: Training and validation accuracy for our three scenarios.

other two scenarios in Top-1 accuracy, it is interesting that the error dispersion affects the keys topologically close to the target one.

In Figure 14, we compare our model and human performance on the key classification task. The misclassification error and the dispersion result are significantly lower for our algorithm. Moreover, it can be noticed how the four keys on which humans perform the best match those in the corners of our keypad (i.e., 1, 3, 7, and 9).

C Additional Experiments

To gain further insight into how coverage can affect the attack performance, we grouped the tested users by the coverage strategy:

- `Side`: The non-typing hand rests on the side of the palm and is angled to cover the keys of the PIN pad (40% of users applied this covering strategy).
- `Over`: The non-typing hand is raised completely off the surface, covering the PIN pad both with the entire back of the hand and the fingers (43% of users applied this covering strategy).
- `Top`: The fingers of the non-typing hand rest on the top of the PIN pad, and the back of the hand is used for the coverage (17% of users applied this covering strategy).

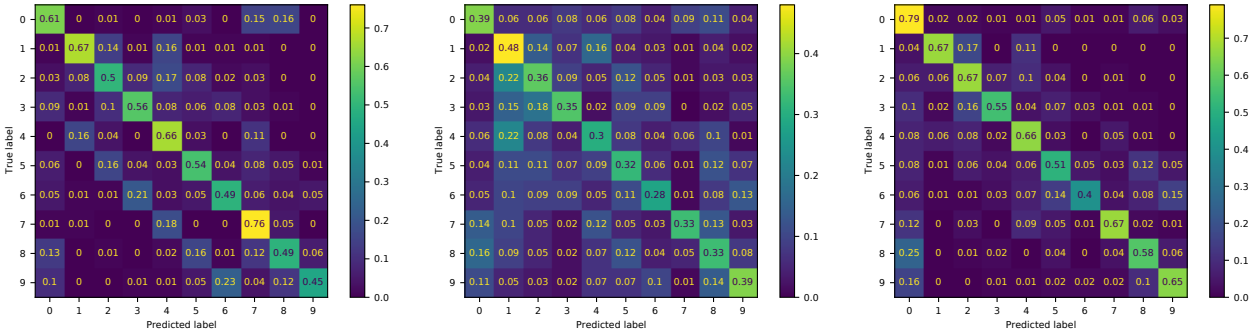
In Table 3, we report key and PIN TOP-3 accuracies for our approach. Clearly, `Side` covering strategy provides the least protection and should be avoided. At the same time, the `Over` and `Top` covering strategies provide much better protection. Interestingly, we see that with the `Over` covering strategy, the *Mixed* scenario reaches lower accuracy than the *Single PIN pad* scenario. We postulate this happens as this covering strategy makes it less "natural" for the user to type, deceiving the deep learning algorithm. Further attack improvements could be made with datasets having examples of one covering strategy only. For the `Top` covering strategy, there were no data for two out of three scenarios (denoted NA in Table 3).

Covering strategy	Scenario	Key accuracy	PIN TOP-3 accuracy
`Side`	Single	0.64	0.30
	Independent	0.42	0.12
	Mixed	0.77	0.53
`Over`	Single	0.52	0.12
	Independent	0.31	0.10
	Mixed	0.46	0.07
`Top`	Single	NA	NA
	Independent	0.41	0.13
	Mixed	NA	NA

Table 3: Performance of our attack for different covering strategies in *Single PIN pad*, *PIN pad independent*, and *Mixed* scenarios. `Top` covering participants were present in the *PIN pad independent* scenario only, as for the others, no data were available (NA).

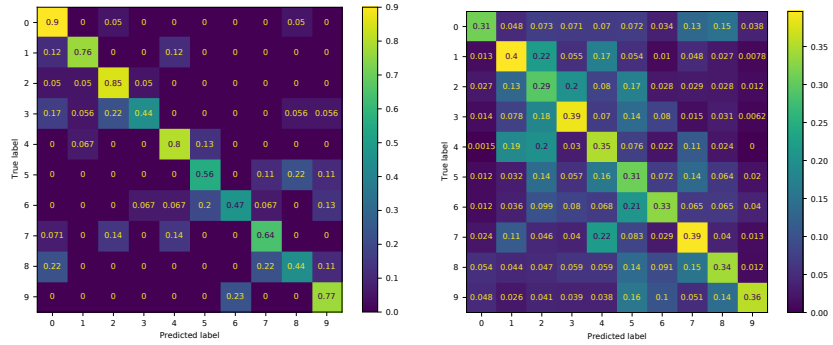
For the PIN shield countermeasure, we depict various levels of hiding in Figure 16. There, 25% denotes that the first row of the PIN pad is covered (simulated with a black patch), 50% first two rows, 75% first three rows, and finally, 100% all four rows of the PIN pad are covered. Note that we do not include the covering with the other hand into these percentages.

Table 4 provides results for several additional attack configurations. First, we performed two experiments simulating a lower camera quality or a larger camera distance from the PIN pad. For this purpose, we reduced the model input resolution from 250 x 250 to 125 x 125 and to 64 x 64. Results show that our model maintains an accuracy higher than 20%, even when halving the input resolution (i.e., doubling the camera distance). However, this is not to be considered as a physical limitation for our attack since if the attacker places a camera outside the ATM chassis, it is possible to use an optical zoom. Further, many pinhole cameras can record with a resolution



(a) *Single PIN pad* scenario. (b) *PIN pad independent* scenario. (c) *Mixed* scenario.

Figure 13: Confusion matrices of key predictions (predicted labels) vs. true values (true labels) for our three scenarios.

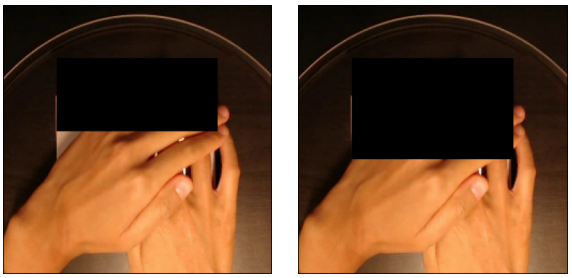


(a) *Recalculated confusion matrix for our algorithm (Mixed scenario).* (b) *Confusion matrix for humans.*

Figure 14: Confusion matrix comparison between our algorithm and humans.

(a) *Left-corner camera.* (b) *Center camera.* (c) *Right-corner camera.*

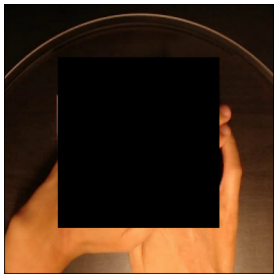
Figure 15: Same video frame recorded by three cameras.



(a) *25% of PIN pad surface covered (i.e., digits form 1 to 3).*

(b) *50% of PIN pad surface covered (i.e., digits form 1 to 6).*

(c) *75% of PIN pad surface covered (i.e., digits form 1 to 9).*



(d) *100% of PIN pad surface covered (i.e., no digit is visible).*

Figure 16: PIN pad shield configurations.

up to 1 080p [9], which is higher than the resolution we used to collect our dataset (720p).

Next, we investigated the accuracy of our attack leveraging different camera positions. In particular, we performed two experiments training and testing our model with the left-corner and the right-corner cameras, respectively. Figure 15 shows the camera views used in our experiments. The results give a significant difference in performance if the camera is on the right or the left. This is because the participants in our experiment were right-handed, and therefore filming from the right had worse coverage of the PIN pad and typing hand. In contrast, the typing hand and the PIN pad were almost completely covered using shots from the left, significantly reducing the model's performance. We also evaluated whether using video from all three cameras in training (the experiment "multi-camera training" in Table 4) could improve the accuracy of our model when compared with videos recorded from the center camera only. The results show a drop in performance, which we attribute to the higher variance in the data provided as input to the model.

Finally, we report the results of our model without data augmentation and without including the blacklisted users in the training set. In both configurations, the performance of our model drops, showing that reducing the training size is penalized heavily. Note that even in the worst case of a camera placed on the left corner (i.e., the one with less visibility), *our model still performs better than an average human.*

[9] https://www.dsecctv.com/Prod_telecamere_spioncino_porta_AHD.htm

Experiment	Key accuracy	PIN TOP-3 accuracy
Input resolution 125 x 125	0.55	0.23
Input resolution 64 x 64	0.47	0.15
Left-corner camera	0.46	0.10
Right-corner camera	0.62	0.31
Multi-camera training	0.53	0.22
No data augmentation	0.44	0.11
Blacklisted excluded in training	0.54	0.18

Table 4: Additional attack configurations and results in the *Mixed* scenario.

Frame error confidence (p<0.01)	Key accuracy	PIN TOP-3 accuracy
3	0.60	0.29
5	0.59	0.26
10	0.54	0.16
15	0.49	0.12
20	0.12	0.06

Table 5: Performance of our attack in the *Mixed* scenario assuming different levels of frame detection error.

In this paper, we used the feedback sound emitted by the PIN pad as a detection system for the frames containing a keystroke. To evaluate the impact of other frame detection systems, we conducted an experiment varying the frame extraction precision. We simulated the detection error by adding Gaussian noise with mean zero to the ground truth (i.e., the frame position in the video). In Table 5, we report the single key and the PIN TOP-3 accuracies for the *Mixed* scenario, simulating five levels of the frame detection error. Compared to the results obtained using the audio feedback (key accuracy 0.61, 5-digits PIN Top-3 accuracy 0.30), we see that our model works well even with small/medium levels of frame detection error (i.e., less than five frames). In particular, for a frame error confidence of three (i.e., when the frames are detected through the appearance on the screen of the masked symbols [8]), the performance drops only 1% both for key and TOP-3 PIN accuracies. Contrarily, when the detection error becomes high (i.e., more than 15 frames), the performance of our model decreases significantly. This happens since the frames considered by the model do not contain information related to the target key, as they are too temporally shifted. Naturally, if the attacker recognizes a situation like this, it would be possible to mitigate the effect of detection error by not using the feedback sound but observing the appearance of "*" symbols on the screen.