Your PIN is Mine: Uncovering Users' PINs at Point of Sale Machines

Stefano Cecconello[1,2], Matteo Cardaioli[3], Luca Pasa[1], Stjepan Picek[4, 5], and Georgios Smaragdakis[2]
[1]University of Padua, Padova, Italy
[2]Cybersecurity group, Delft University of Technology, Delft, The Netherlands
[3]GFT Italy, Padova, Italy
[4]Faculty of Science, Radboud University, Nijmegen, The Netherlands
[5]University of Zagreb Faculty of Electrical Engineering and Computing, Unska 3, 10000 Zagreb, Croatia

***Abstract*—Point of Sale (PoS) machines have become extremely popular recently. In many economies, most transactions occur using them. Although PoS technology is evolving, PINs are still heavily used. In this paper, we perform a large-scale study to understand how difficult it is to uncover user PINs at PoS, even when the users cover the pad with their hands. Our study involves 142 participants, two types of PoS, and around 13,800 PINs. We develop machine learning techniques to infer PoS PINs by using hidden cameras. Our results show that uncovering PINs in PoS is more complex than in other cases where a user PIN is used, e.g., ATMs, because of the small pad area of PoS. Nevertheless, we could achieve more than 50% Top-3 accuracy for 4-digit PINs and 45% Top-3 accuracy for 5-digit PINs, even when the PIN is covered by the user's hand. We comment on the impact of the camera's position and PoS on the successful inference of the user's PINs. We also comment on the hardness of inferring PINs depending on the physical distance of digits and recommend what are good practices to generate PINs and cover PoS to make PIN inference difficult.**

***Index Terms*—Authentication, Security and Privacy Protection, Machine Learning, Usable Security.**

I. Introduction

A Point of Service (PoS) machine is a device that interfaces with payment cards to make electronic money transfers. Based on recent studies of PoS transactions in terms of value, the share of card transactions in 2022 was higher than that of cash transactions for the first time [1], [2]. Transaction value in the PoS payments market was US$11.6 trillion in 2021 alone [3].

Despite the convenience of using PoS and credit cards for online payments, serious threats exist. Cloning a payment credit card is easy [4], [5]. The inference of a payment card's Personal Identification Number (PIN) is the second step to fully impersonating a credit card for criminal activities [6], [7]. It is worth mentioning that in many regions like North America, Europe, Asia, and Australia, the use of PoS is either the most popular payment method or the default payment method, e.g., to improve efficiency or for the collections of taxes, e.g., Value Added Tax (VAT). At the same time, PoS has undergone many technological improvements, such as supporting contactless cards, but it is still common to require a PIN. This is the case when a high-value transaction has to be authorized or when a particular limit has been reached in the card balance. In some European countries, transactions that cost more than 50 euros require a PIN [2]. Moreover, it is also possible to disable the contactless option with the Prilex malware [8]. Indeed, it can block contactless near-field communication (NFC) transactions on infected devices, which forces customers to use their physical credit cards.

While previous works have investigated the inference of PINs in Automatic Teller Machines (ATMs) [9] and mobile phones or iPads [10], [11], side-channel attacks' success in inferring the PIN in PoS is not studied. In this paper, we perform a large-scale study with more than one hundred twenty participants who enter PINs in different PoS to assess how easy it is to infer PINs in various settings. This includes the cases in which cardholders follow best practices, i.e., to enter their PIN with covered hands.

For our study, we must deal with unique challenges connected with the attack on the PoS PIN mechanism. First, we have to develop new and complex deep learning models to guess the PINs. Due to the small size of the PoSs, the models are more complicated than what was proposed, for instance, for the attack on the ATM PIN mechanism [9]. Second, our attack leverages video frames, i.e., images. Previously proposed techniques that were shown to be effective for inferring ATM PIN [9] have never been proven to be efficient in the PoS setting.

On the practical side, PoSes are typically small and have a limited surface area, making attacks difficult. For instance, the keys are commonly much smaller for PoS than for ATM, and the distance between the keys is commonly much smaller for PoS. Moreover, there are many different PoS layouts and sizes. Thus, the success of the attacks may vary. Another aspect to consider is the versatility of PoS usage. Unlike other systems, a PoS is not always static; it can be held or used at various angles, introducing a range of potential security vulnerabilities. For instance, the device could be used at a slope or held at different distances from a camera, necessitating a comprehensive security approach.
Our contributions can be summarized as follows:

- We performed a large-scale study with 142 (125 unique) participants collecting video captures of more than 13,800 PINs entered in two different Point of Sale (PoS) devices.
- We introduce a new attack to infer PoS PINs. We show that our attack is effective despite the small PoS surface. The attack is also effective for various types of PoS machines, PIN tilted positions, and different styles of PIN hand covering.

- We show that it is possible to achieve 50% Top-3 accuracy for 4-digit PINs and 45% Top-3 accuracy for 5-digit PINs, even when the PIN is covered by the user's hand for the three different scenarios we considered.
- We analyze how the design and layout of PoS devices impact the inference of PoS PINs. It is more difficult to infer PINs in PoS with a smaller pad surface, and it is easier to infer PINs when users hold the PoS at an angle.
- We investigate if there are pairs of consecutive digits that are more or less secure than others. Our results show that the physical distance between PIN digits plays an important role. We provide recommendations to strengthen PoS PINs based on the insights derived from our study. In all the settings we evaluated, the digit pair (9, 7) was inferred with high accuracy. In stark contrast, the digit pair (5, 5) was inferred with the lowest accuracy.
- We examine the impact of the distance between a target key and the key pressed before it on PIN inference accuracy, defining Euclidean distances for key pairs. This analysis examines both horizontal and vertical movements, enabling us to evaluate how various key arrangements affect vulnerability. We also consider the direction of movement, revealing valuable insights into how key layout and typing styles influence the effectiveness of inference attacks on PINs.
- We show that the distance between the camera and the PoS plays an important role. Indeed, when the camera is up to 4 meters above the PoS, the PIN inference accuracy is very high. When the distance between the camera and the PoS is 8 meters or higher, the PIN inference accuracy drops below 10% for most of the settings we considered.
- We will make the artifacts of this work publicly available, including the anonymized collected data and our code, to improve the generation of PIN codes that are used in PoS and to enable future work in this research area.

II. Related work

In the last decade, the tracking and classification of hand and finger positions/movements have been extensively investigated in computer vision due to applications to fields ranging from Human-Computer Interaction (HCI) [12] to training and rehabilitation in medical applications [13]. Among the first approaches were solutions based on Convolutional Neural Networks (CNNs) [14], [15], where a hierarchically-decreasing pipeline of the convolutional layers is used to generate a set of shape features that are finally classified by the last fully-connected stages into a static hand pose. On the other hand, some other solutions combine traditional image pre-processing steps aimed at cleaning and refining the image quality before classifying the resulting image with the deep learning detection architecture [16]. In other strategies, using contexts allows the network to generate separate manifolds in the training phase [17].

Multiple works consider the security of systems where communication with them is established through typing/hand movement. Commonly, those works use various side channel information, making them implementation attacks. We note that implementation attacks are not limited to authentication mechanisms but are also employed in, e.g., cryptography with timing [18], [19], power [20], electromagnetic analysis [21], sound [22], and temperature [23] side channels.

Song et al. [24] measured the timing between keystrokes in interactive Secure Shell Protocol (SSH) sessions in an attempt to retrieve the typed passwords. Considering attacks on ATMs, Balagani et al. [25] recorded the screen of an ATM while the victim entered the PIN. The timing information used was the PIN masking symbols appearing on the ATM screen, and from there, inferring which keys were most likely typed by the victim. Liu et al. [26] considered user-independent inter-keystroke timing attacks on PINs. The attack is based on an inter-keystroke timing dictionary built from a human cognitive model.

One of the first works considering hand movement as a side channel was done by Balzarotti et al. [27]. There, the authors investigated the hand movements of people typing on a keyboard and then reconstructing the text being typed. Shukla et al. [10] considered the finger motion during the PIN-entry process on smartphones and showed that 50% of 4-digit PINs can be retrieved in one attempt. This work has a similarity with one of our attack scenarios where the PoS machine is set at 45° angle as the person uses only one (known) finger to type. Sun et al. [28] investigated attacks on tablets where they considered the backside movements of the tablet as side channel information to deduce what is being typed. Ye et al. [11] showcased an attack to infer the pattern lock of mobile devices from videos. Still, they needed to observe the user's fingertip for the attack to be successful. Besides hand movement, researchers also showcased how eye movement can be used. For instance, Cazorla et al. [29] considered eye movements observed during video calls to predict text being typed on a keyboard.

Mowery et al. [30] used a thermal camera to attack ATM PINs, concluding it is possible to obtain enough codes to make the attack economically viable. Kaczmarek et al. [31] used thermal information to reconstruct what is being typed on keyboards, where the authors showed results for a number of typical keyboards. While such attacks are relatively powerful and do not require any action during the typing, they rely on the heat signature being present long enough to be recorded after the typing. Compagno et al. [32] used sound to deduce what is being typed. More precisely, the authors showed how having an active Skype call transfers enough information about the text being typed to reconstruct it.

It is also possible to conduct a more powerful attack by considering more than one side channel source. Cardaioli et al. [33] used the ATM's sound when a button is pressed. From there, this information could be used to extract a timestamp of the keys being pressed. Moreover, if such information is combined with a heat signature from a thermal camera, the performance of the attack can increase even more. On the other hand, Cardaioli et al. [9] demonstrated how attacks on ATM PINs, where the side channel is hand movement, can outperform previous approaches even if the user is hiding the typing hand with the other hand. This approach relied on a carefully designed and trained deep learning model.

Some of the more recent works also consider novel side

channels. For instance, Jin et al. [34] considered human-coupled electromagnetic emanations from touchscreens to infer sensitive inputs on a mobile device. Hu et al. [35] provided a keystroke inference method to eavesdrop on keystrokes on smartphones. The approach is based on eavesdropping beamforming feedback information (offered by the latest Wi-Fi hardware). Ni et al. [36] designed a contactless and context-aware wireless-charging side-channel attack, allowing them to guess screen-unlocking passwords or application launches.

Our work differs from the previous ones in several key aspects: We consider PoS devices, which, due to the different shapes and positions in which they can be set, resemble attacks on mobile phones more than attacks on ATMs. Our attack considers multiple side channels that do not depend on specific hardware. Indeed, we require timing information for key press detection (since sound is not available as a side channel) and hand movement for key guessing. Both necessitate recordings with commodity cameras only. We consider the case where the user hides the PIN entered by the other hand.

III. PoS Attack

Assumptions: We assume the victim interacts with a generic PoS, performing PIN-based authentication. We assume that the victim is a careful user who adopts techniques to improve their security. In particular, while entering the PIN, the user tries to cover the view of the keypad with his free hand. The attacker aims to learn the victim's PIN by placing a camera near the PoS. The attacker can also use a compromised service camera in the shop if pointing to the PoS with a good viewing angle. The PoS system is equipped with a small monitor that displays obfuscated symbols when users enter a PIN, obscuring the entered digits with asterisks. We do not assume that the PoS or its PIN pad have been compromised during the attack. The camera stores recorded video. How the video is stored is irrelevant to our attack, i.e., it can be stored locally or offloaded to a remote site. Our attack relies only on that recorded video. No audio is recorded from the webcam. We assume that the PoS emits no feedback sound when pressed.

Previous attacks (see Section II) have shown it is possible to recover information about PIN typed on a keyboard where the attack relies solely on audio. However, in our case, this type of attack is not applicable. In fact, (i) the distance between the (microphone of the) webcam and the keypad is at the limit according to the previous experiments conducted, (ii) the rubber keypad used by PoS drastically reduces the sound produced when the keypress is pressed, and (iii) in [9], the experiments are conducted on an ATM, and these devices are often placed in secluded places that are usually quiet and allow to collect audio with low noise.

We assume that the refresh rate of the PoS screen is at least 30 Hz, which is a widely acceptable threshold, considering that the normal refresh rate of a screen is 60 Hz. This rate equals the number of frames per second of our videos and allows us to collect high-accuracy timestamps (see Section IV-B).

Preliminaries: To set up our attack, the attacker must select a target PoS and hide a webcam nearby. An optimal position for the webcam is to place it vertically above the PoS, ideally on the ceiling or anchored to some structure near the PoS. The precise positioning of the webcam may differ. Still, in our attack scenario, we placed the camera vertically over the PoS at a distance of 1 meter (we also simulated higher distances as explained in Section V-C).

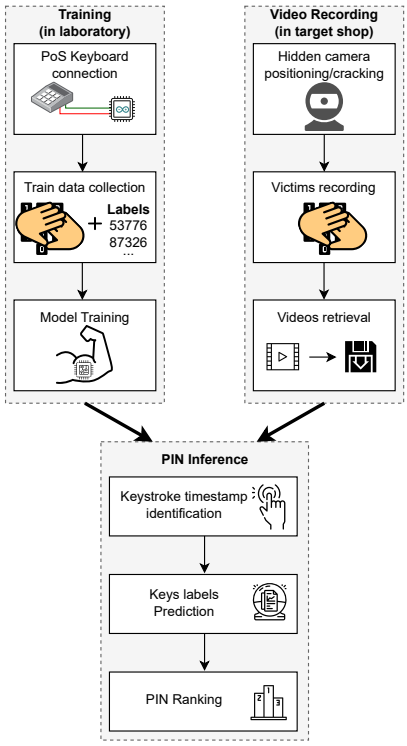


Fig. 1. Our attack setup consists of three phases: model training, video recording, and PIN inference.

As shown in Figure 1, our attack consists of three phases:

1) Training (discussed in Section III-A).
2) Video Recording (discussed in Section III-B).
3) PIN Inference (discussed in Section III-C).

A. Training Process

The attacker initially selects a target PoS model. Next, they must find a copy of the target PoS to collect the data necessary to train the neural network. In general, this search is straightforward, as second-hand PoS models are typically readily available for online purchase. Additionally, given that, in any case, the PoS must be modified (as described below), the attacker may opt to acquire a non-working PoS. This streamlines the search further, as defective PoS units are frequently sold to recover spare parts.

The attacker uses the PoS copy to assemble the training dataset. During the data collection, they insert sequences of PINs mimicking the victim's actions when entering the PIN (e.g., obscuring the hand while typing). During data collection, the attacker extracts two types of information: (i) the digits entered and (ii) the timestamp of the keypress. These data are necessary to define the ground truth of the dataset and to simplify the extraction of the frames given as input to the neural network, respectively. Since the PoS does not provide this kind of information, the attacker must modify

the internal hardware by substituting the internal PCB with a personalized one. Further details about the manipulation of the PoS are available in Section IV-B. We point out that the PoS manipulated in this phase is used to collect training data, while the target PoS(es) of the attack do not undergo any modification.

Finally, the attacker trains the predictive model using the collected training dataset. For an in-depth examination of the implemented model, we direct readers to Section IV-F. Note that the length of the collected PINs is a constant factor for the network. Therefore, the attacker will have to train different networks if they want to attack PINs of different lengths.

B. Video Recording Process

In this phase, the attacker hides the camera close to the target PoS. The camera records the victims while typing their PINs. Many spycams are available online. The small size of these devices makes it possible to hide them easily. The attacker can select a target PoS based on the simplicity of hiding the webcam more easily. These devices have a low cost (between 15-80€) that changes based on available functions. The recording quality can reach up to 4K resolution. The battery can last for 6-8 hours, and some cameras also provide a motion sensor, which can sensibly increase the duration of the battery by turning on the registration only when a movement is detected. The data can be saved in a local memory or a remote device wirelessly. Finally, the attacker retrieves the recorded video.

C. PIN Inference Process

The goal of this phase is to infer the PIN entered by the victim. The first step consists of retrieving the timestamps of the keypress, which are necessary to give the right frames in input to the trained model. The timestamp can be retrieved by taking advantage of the masking symbols appearing on the screen while the victim is typing. Note that the attack set timestamps are not the same as in the training set, since for the latter, the timestamps are collected directly from the PoS copy (further details in Section IV-B). Thanks to the timestamps, the attacker creates a new collection of frames called an attack set. This set consists of N groups of M frames each, where N is the number of collected digits and M is the number of frames for each digit.

Next, we provide the attack set as the input to the trained model. For each one of the N groups, the model returns the likelihood of each class (i.e., the 10 digits) to be the one corresponding to the input sequence of M frames. At this point, the attacker must group the network outputs in subsets of 4-5 (based on the lengths of the original PINs entered by the victims). This operation is very simple, thanks to the collected timestamps. Finally, leveraging the predictions contained in a mini-set, 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.

(a) *Ingenico Model iPP 320* (b) *Ingenico Model iPP 220*

Fig. 2. The real PoS PIN pads used in the data collection. The two PoSs are depicted using the same scale. Thus, the relationship between their dimensions is realistic.

IV. Experimental Setting

A. Investigated PIN Pads

We perform three data collections using two PoS PIN Pad devices: Ingenico iPP 320 (Figure 2a) and Ingenico iPP 220 (Figure 2b). We selected this brand since it is the larger manufacturer based on the statistics of the last years [37].

Although the two devices' brands are the same, we selected them because they have been designed for different purposes. Model *iPP 320* is a "regular" PoS, and it provides the possibility to insert the card, use the contactless, or swipe the card to pay. While entering the PIN, the shop operator can hold the PoS in their hand or place it on a flat surface. However, the manufacturer also provides a stand that allows the PoS to be kept fixed with an inclination of approximately $45°$ to the surface. The support can rotate to facilitate data entry for both the user and the operator. Model *iPP 220* is a PoS "extension". It must be used in combination with a regular PoS (e.g., iPP 320) to which it is connected via a cable. Its main feature is its small size, as it has been designed as an external keypad that the operator can hand to the customer to make it easier for them to enter the PIN. During the insert PIN phase, it is usually placed on a surface.

Below, we describe the differences between the two devices in detail:

- *iPP 320* has a maximum dimension of 82 mm $\times$ 170 mm, while *iPP 220* is much smaller with a size of 74 mm $\times$ 130 mm (i.e., approximately 25% less length on the vertical dimension).
- The buttons of *iPP 320* have a size of 17 mm $\times$ 9 mm, while the buttons of *iPP 220* have a size that varies in the range of 12-13 mm $\times$ 6 mm.
- The horizontal distance between two buttons is 4 mm for *iPP 320* and 3 mm for *iPP 220*, while the vertical distance is 4 mm for *iPP 320* and 2 mm for *iPP 220*.

During the data collection, we placed the PIN pads on a table at a height from the ground of 115 cm and a distance

TABLE I
THE STATISTICS FOR THE THREE DATASETS.

	Total	Male	Female	Non-binary	Age	Full PINs
iPP 320	59	44	14	1	24.47 ± 4.04	5,769
iPP 220	41	27	14	0	23.41 ± 5.57	4,015
iPP 320 45°	42	29	12	1	24.12 ± 5.80	4,066
All datasets	142	100	40	2	24.06 ± 5.05	13,850

from the border of the table of 15 cm. We opted for such a setup to emulate the common position (height) of a PoS. We investigated three different configurations for the positioning of our two PoSs: *iPP 220* has been placed horizontally to the table, while *iPP 320* has been placed both horizontally and $45°$ to the table (to simulate the stand provided by the manufacturer). To perform the video recording, we used a Logitech Brio Ultra HD PRO anchored to a tripod. We placed the webcam perpendicular to the table at a distance of 1 meter from the PIN pad. When the device was inclined $45°$, we measured the distance from the center of the PIN Pad (i.e., between numbers 5 and 8). We used this criterion since when the PoS is inclined $45°$, the distance from the camera to the PIN pad changes by centimeters, considering the higher or lower part of the PIN pad. The center of the PIN pad provides an average distance of 1m. To prevent the POS from moving outside the webcam's field of view, we anchored the POS to the table. Possible POS movements have been simulated in the regularization phase described below in Section IV-F. Video acquisition was performed using OpenCV's `VideoCapture` to interface Python with the Logitech webcam. The camera was configured for 1080p resolution (1920×1080) at 30 fps, using the MJPG codec and autofocus disabled. Frames were captured in BGR format and later converted to grayscale during pre-processing (see Section IV-D). The Brio's automatically managed exposure and contrast, and the field of view was set to its balanced 78° setting.

B. Data collection

We performed three data collections. The statistics related to the data collected are reported in Table I. We also reported the total statistics for the sum of all datasets. The column "Full PINs" reports the number of 5-digit PINs used for the experiment. The total number of PINs entered is 14,200 (i.e., 142 participants $\times$ 100 PIN per participant). However, we excluded some PINs containing typos, whereas, for typos, we mean PINs with a number of digits different from five. The final number of PINs is 13,850. Among the 142 participants who participated in the experiment, 125 were distinct. This overlap does not pose a problem, as in the experiments conducted, data from the three data collections are not used together.

All sets of data consist solely of right-handed participants. All participants provided their consent for data collection and usage by signing an informed consent form. The authors of this paper have anonymized all data and utilized it exclusively for research purposes. No private information about the participants has been stored. We instructed participants to type some random PINs before starting the experiments to help them feel more comfortable with the device and to decide which typing style they preferred (e.g., using one finger or multiple fingers).

Fig. 3. The PoS iPP320 shown in the $45°$ setup. In the figure, we can observe the stand that we designed and 3D printed to keep the PoS in the correct position.

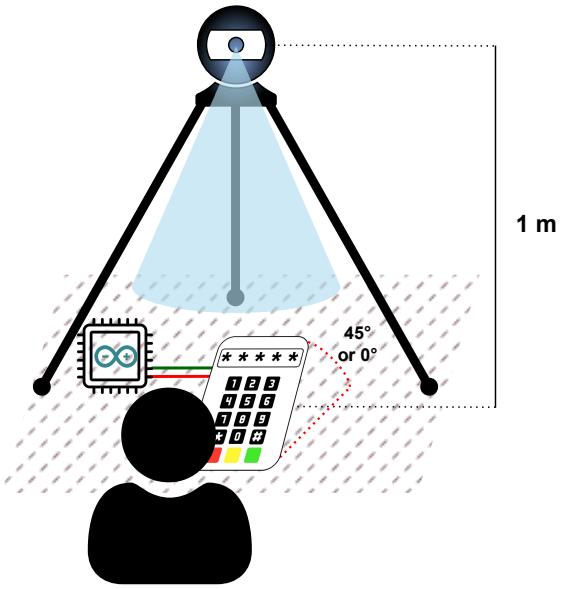


Fig. 4. Our testbed configuration for the experimental setup. The webcam was placed on a tripod at a distance of 1m. The ground truth was saved using an Arduino connected to a computer.

In the third data collection, we constructed a support to keep the PoS inclined at $45°$ with respect to the table. We designed it using a 3D computer graphics application, thus making it compatible with the connection for the real stand provided by the manufacturer. Our support has been developed so as not to come into contact with the participant while typing, and thus not to change their writing style. Finally, we used a 3D printer to print the final support used during the experiments. In Figure 3, we show the final version of our 3D printed support.

During the third data collection, it is noteworthy that all participants autonomously decided to enter their PINs using their thumbs. Indeed, to feel comfortable typing on a keyboard inclined $45°$, the participants autonomously decided to steady their right hand on the POS to type more accurately. At this point, the thumb was the only finger left available for typing. We did not instruct the participants to type PINs in any specific

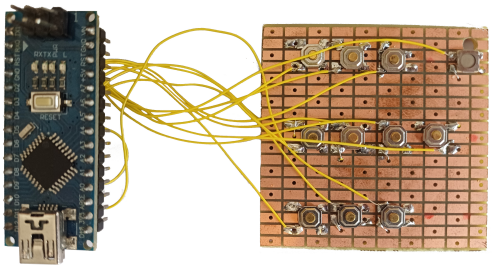


Fig. 5. The picture depicts one of the PCBs we used to perform our data collection. On the left side is the Arduino that we used to collect the keypress, and on the right side is the PCB created for iPP 220, with buttons following the layout of the PIN pad numbers (plus the enter).

way, e.g., by demonstrating how to enter a PIN. Moreover, during the data collection phase, only one participant was allowed to be in a room at any time to avoid a participant being influenced by other participants.

Participants were instructed to position themselves in front of the test PoS and cover their typing hand while entering the PIN during the experiment. They were given autonomy in choosing their covering and typing approach. This procedure aimed to simulate a participant safeguarding the PIN, thereby deterring potential shoulder-surfing attacks. Each participant entered 100 randomly generated 5-digit PINs distributed across four sets of 25 PINs each. This segmentation into four sets was implemented to incorporate brief intermissions in the experiment and prevent participant fatigue. The PINs were presented sequentially on a screen, and the participants were required to press the enter button after inputting each PIN to proceed to the next. A total of 13,850 (see Table I) random 5-digit PINs were recorded. Our study aims to reconstruct the PIN from video sequences independent of participant typing habits or familiarity with the PIN or PIN pad. Therefore, we opted to randomize PINs instead of having participants input the same PIN repeatedly. This approach widens the scope of the attack, allowing it to be applied to mnemonic PINs and One-time Passwords (OTPs).

To carry out the supervised training part of our attack, it was necessary to collect the ground truth about the data entered by the participants. In particular, the information we needed to collect is (i) the instant when a key is pressed, (ii) the instant when a key is released, and (iii) the digit corresponding to a key that has been pressed. For obvious security reasons, PoSes do not provide this information, so to obtain this information, we had to replace the internal components of the PoS with new PCBs. Specifically, we mounted switches on blank PCB boards, arranging them exactly below the PIN Pad keyboard keys. To simulate an experience as similar as possible to that of the original PoS, we tried different keys, selecting those with an operating force (i.e., the force needed to press the button) closest to the touch of the original ones. We inserted the boards inside the PoS and connected the switches to an Arduino, which, via USB, sent the necessary information to a computer to obtain ground truth. Figure 5 shows the PCB used for iPP 220.

The video recordings were synchronized with the timestamp of the key events, providing the foundational data for our experiments. Note that the data collection is done by the attacker "at home", ensuring the attack is realistic. Indeed, we collect this "synchronization" signal to be able to extract the frame from the video automatically. This step is substituted by the detection of the asterisk on the PIN pad screen during the actual attack. Since we are collecting video at 30 fps, our precision is 1/15 second for the sampling theorem (i.e., up to ± 1 frame of error for the timestamp). In the Appendix, we conducted an experiment to show that up to an error of ± 3 frames, the performance of our attack is not significantly affected. The dataset will be provided upon request.

C. Ethical considerations

All the participants in the experiment were informed about the scope of the experiment and provided their explicit consent. To protect the participants' privacy, we did not store the name, date of birth, identification card number, or any other personally identifiable information. We made sure that the camera recorded only the hands of the participants. Before recording, we made sure that there was no tattoo, ring, or other mark that could be used to identify the participant. As there is no information related to human subjects involved, an IRB was not relevant. The participants were free to declare their age and gender.

The PINs were randomly generated and presented on a monitor in front of the participants. We did not use any PIN that is associated with any of the participants' PINs. We did not give any hints on how to cover the PIN tab. The participants were free to type the PIN as they wished.

D. Pre-processing Video

In this section, we describe how we preprocessed the videos after the data collection. This entails executing a series of prescribed procedures on each frame of the video, which are as follows: (i) Transformation of the video frames into grayscale; (ii) Standardization of the input to ensure that all pixel values are within the interval [0, 1]; and (iii) Cropping of the image to a size of 260 $\times$ 300 pixels centered on the PoS. Note that the standardization statistics depend on the training, validation, and test split explained in Section V-A. The parameters are calculated on the training set and applied to the validation and testing sets. The cropping size was set so that the movement of the hands was never cut out of the frame. Therefore, the cropping size is the same for all datasets and does not depend on the PoS size. The next step consists of extracting the portions of the video containing a single keypress. This information is required to train our supervised network. For the training phase, we could use the ground truth provided by the PCB inserted in the PoS. However, this procedure cannot be reproduced in the attack phase since the target PoS is not compromised. For this reason, we assume that the attacker can use the PoS display as a source for this information. Although the user is covering the PIN pad with his left hand, the display is almost always visible in our dataset. Therefore, the asterisk that appears on the display when a keypress is pressed can be used as a time

marker for the attack phase. Once we identified the exact frame corresponding to the keypress, we had to decide how many frames to extract around the target frame. A fixed number of frames is indeed necessary to optimize the neural network training. To ensure uniform input dimensions for our model, each sample was constrained to a fixed sequence length. This length parameter represents an important tradeoff between how much information is given to the model and the risk of overlapping with the previous/following keypress, providing the model a wrong information. To decide the final length, we conducted a statistical examination of the dataset keypress length, in which outlier sequences were excluded by filtering out those with lengths beyond three standard deviations from the mean. The average sequence length, after this filtering, converged around 11 frames (similarly to [9]); therefore, we adopted this length for our input. Finally, we had to manage the edge cases in which the keypress is too close to the end or beginning of the video, and consequently, less than five frames are available before and/or after the target one. We decided to pad the sequence with black frames, but always ensured the target frame remained in the center of the new padded sequence.

E. Machine Learning Setup

For our experiments, we used a machine equipped with an AMD Ryzen Threadripper 2950X 16-Core Processor, 48 GB of RAM, and two GeForce RTX 2080 Ti, where each GPU has 12 GB of RAM. We used PyTorch 2.0.1 and Python 3.10.10 to implement the machine learning models.

F. Investigated DL Models

Our attack aims to predict the key pressed by the victim despite them covering the PoS with the non-typing hand. The data we use to achieve this objective is a video that shows the user's hands at the moment of entering the PIN. Based on these initial assumptions, we've crafted a robust deep learning model that effectively encompasses both the spatial and sequential dimensions of the input by integrating Convolutional Neural Networks (CNN) and Long Short-Term Memory layer (LSTM) layers. Such models are recognized today as one of the best state-of-the-art tools for the analysis and classification of video data [38]. Therefore, our neural network operates in parallel, processing multiple frames simultaneously. The specific number of frames is meticulously selected through a comprehensive validation process, ensuring performance and efficiency. The initial layer of our model harnesses independent CNNs for each frame. The outputs of the CNNs are then concatenated and sent to the recurrent network, which processes them as a sequence of timesteps. Finally, the output of the recurrent network is projected into the output space.In the training process of the resulting model, we divide our dataset into training, validation, and test sets. All hyperparameters undergo rigorous validation, with model selection conducted through a randomized grid search during the validation phase.

More precisely, due to the high time requirements of performing an extensive grid search, we decided to limit the number of values taken into account for each hyperparameter by performing some preliminary tests. Specifically, for CNNs, we experimented with kernel sizes of 3×3, 6×6, and 9×9 while varying the number of convolutional layers within the range of 2 to 4 and the stride in the range 1 to 2. Subsequently, we employed a combination of dropout and batch normalization for regularization. Within the dropout layers, we explored dropout rates ranging from 0.01 to 0.2. We also explored the use of batch normalization, in particular PyTorch BatchNorm2d in convolutional layers and BatchNorm1d after flattening. For the Long Short-Term Memory (LSTM) layers, we investigated configurations with 1 to 4 layers and unit sizes of 16, 32, 64, and 128. In the case of the layers used to project the LSTM output in the output space, we explored a funnel architecture characterized by decreasing units in subsequent layers, with each successive layer having half the units of its predecessor. We explored a unit size for the first layer from 64 to 256.

To address differences stemming from different positions (tilt, distance) of the camera or PoS, we also apply regularization with noise to the training data [39]. Such measurements then encompass a broader range of scenarios to mitigate the variations. Specifically, we applied the following video-based transformations:

- Rotation, with a maximum angle of 7 degrees in both clockwise and counterclockwise directions.
- Horizontal shift, with a maximum displacement of 10% of the width.
- Vertical shift, with a maximum displacement of 10% of the height.
- Zoom, ranging from 0.9 to 1.1. Synthetic samples were created by randomly applying combinations of the aforementioned transformation techniques.

To prevent overfitting, we applied early stopping based on the validation loss with a patience of 6 epochs, halting training when no improvement was observed for 6 consecutive epochs. The model was trained using a learning rate of 0.05 and a batch size of 16, which were selected based on preliminary experiments to balance convergence speed and stability.

V. Experimental Results

A. PoS Attack results

We investigated the results of our experiment for the three configurations described in Section IV: PoS iPP 320, PoS iPP 220, and PoS iPP 320 $45°$. We implemented a participant-independent split strategy. This ensures that videos from a given participant are included in only one set among training, validation, and testing. This approach enables a more realistic evaluation of the model's performance since, in a real attack, videos from the victims are not available for the training and validation procedure. Our focus on evaluating the accuracy of PIN reconstruction also led us to exclude any sequences containing fewer than 5 digits (resulting from participants' typing errors when entering PINs). To make our evaluation of model performance as realistic as possible, we also checked the videos in our dataset and excluded participants who did not cover sufficiently during the typing phase. The procedure for

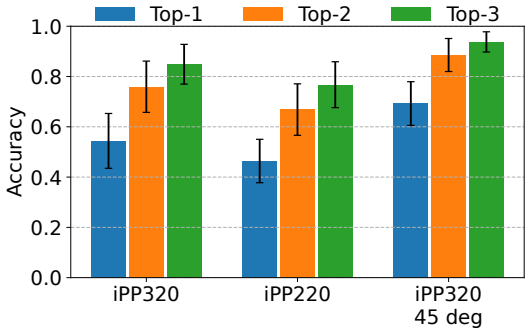
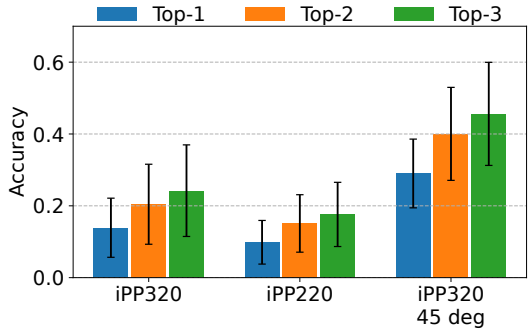
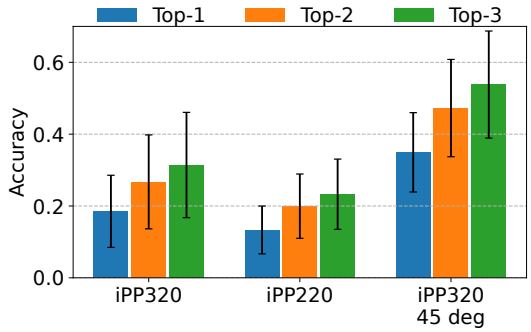


Fig. 6. Single key accuracy of our algorithm for the three considered scenarios. Top-N means that we guessed the digit within the N attempts.



(a) *5-digit PINs.*



(b) *4-digit PINs.*

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

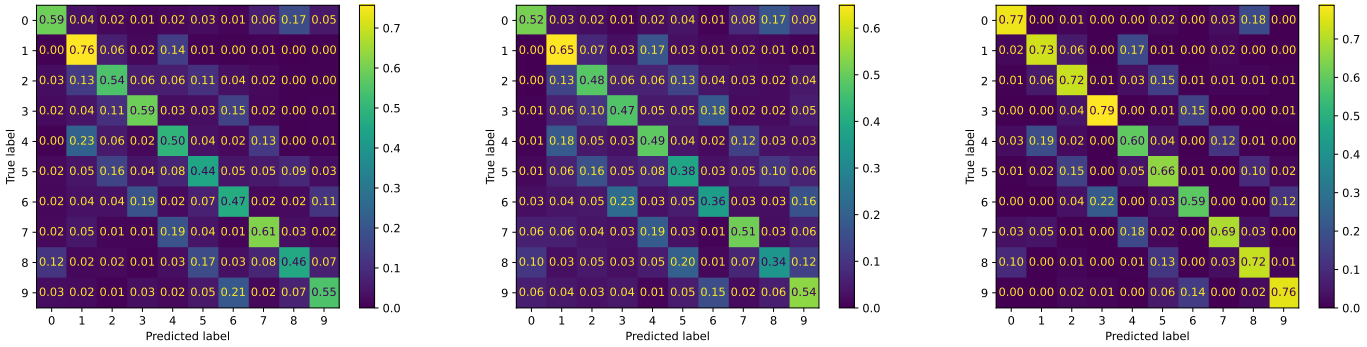
selecting participants who did not cover adequately follows the one from [9]. We created three blacklists of five, eight, and four participants from the first, second, and third data collection, respectively. The data collected from a participant included in a blacklist have been used only in the training set but not in the validation or test sets. We tested the results by performing a 10-fold cross-validation. The training set was then divided into training and validation, ensuring that the validation size was approximately the same as the test size (in terms of the absolute number of samples).

In Figure 6, we report the accuracy for single key classification for all three datasets. The graph shows the Top-1, Top-2, and Top-3 accuracy. We report the first three Top-N accuracy since they are all linked to the calculation of the accuracy of the PINs, even if with decreasing importance. The results show that even in the worst case, corresponding to PoS iPP220, the accuracy remains well above random guessing. The lower accuracy for this dataset is in line with expectations, as PoS iPP 220 is smaller than iPP 320. This difference in size implies less movement of the hands during typing, making extracting significant features by the trained model more complicated. Another factor to consider is the coverage quality, which increases significantly since a smaller PoS is much simpler to cover. The most interesting scenario occurs with the dataset for PoS iPP 320 $45°$, which shows higher accuracy than the other two scenarios. This result can be associated with the uniform typing style used by the participants of this dataset. Indeed, by performing the training procedure with only one typing style, the DL model can specialize better on the assigned task, thus achieving better performance.

In Figure 7, we report the results for 5-digit and 4-digit PINs. In this case, the most interesting metric is the Top-3 accuracy since the PoS allows three attempts before blocking the card. The results are in line with the single key accuracy. For 5-digit PINs, the lowest result is achieved by PoS iPP 220 with a Top-3 5-digit PIN accuracy of 16,8%, which can still be considered a good result. Interestingly, we observe that the variances for all datasets are rather relevant compared to the corresponding accuracies. This is due to the variance in the covering quality of the participants. Indeed, even if we removed the participants who were not covering the typing hand properly, the covering quality was still not the same among all participants. For 4-digit PINs, the results follow the same trend as 5-digit PINs.

In Figure 8, we reported the confusion matrices for our three datasets. We immediately noticed that the spatial disposition of the keys is important. Indeed, prediction errors are often made with keys close to the true one. An example visible is the digits 4, which is confused with the keys 1 and 7, or the key 0, which is confused with 8. In this regard, iPP 320 $45°$ is the dataset that obtains the best results. However, the spatial error is still clearly visible. We noticed that the improvement in accuracy for iPP 320 $45°$ is attributable to a reduction of the error for keys spatially far from the target key. These observations highlight how specific PoS design elements — such as keypad layout, key spacing, and physical inclination — significantly affect the difficulty of inferring the correct PIN. As such, PoS design must be considered a critical factor in the resilience of the system to side-channel attacks.

In addition to Top-N accuracy, entropy provides a precise measure of how much uncertainty remains in the model's predictions. Our model outputs a full probability distribution over the 10 possible digits for each of the 5 PIN positions, allowing us to compute the average entropy of these distributions across the test set. Lower entropy reflects sharper predictions — i.e., the model concentrates probability mass on fewer likely digits. For context, a uniformly random 4-digit PIN has a maximum entropy of $\log_2(10^4) \approx 13.29$ bits, while a uniformly random 6-digit PIN has $\log_2(10^6) \approx 19.93$ bits. In practice, human-chosen PINs are far less random: Wang et al. [40] report an average entropy of 8.41 bits for 4-digit PINs and 13.21 bits for



(a) iPP 320 (b) iPP 220 (c) iPP 320 45°

Fig. 8. Confusion matrices of key predictions (predicted labels) vs. true values (true labels) for our three datasets.

6-digit PINs. By contrast, after our attack, the entropy of the model's predictions on 5-digit PINs is reduced to 3.79 ± 0.92 bits for iPP 320, 5.52 ± 1.58 bits for iPP 220, and as low as 2.42 ± 0.90 bits for iPP 320 at 45°. These post-attack values are not only far below the theoretical maximums for uniformly random PINs but also substantially lower than the entropy of real-world, human-chosen PINs. Thus, our side-channel attack extracts more information about the PIN than would be available by exploiting human selection bias alone, underscoring its effectiveness and the serious security risk it poses.

B. Results Based on Participants' Covering Style

Next, we evaluate our method's accuracy based on the coverage style of the victim. Since the covering methods can be analyzed according to multiple criteria, we analyzed all dataset videos, observing the covering styles kept by the participant while typing. This analysis has been conducted independently by two authors to avoid a single person biasing the classification. Based on the analysis conducted, we identified two criteria that can be associated with the covering style of our participants. These criteria are related to (i) the position of the covering hand on the PoS and (ii) the movement of the covering hand. The first criterion refers to the different ways of positioning the covering hand adopted by our participants. The second refers to the mobility of the covering hand.

For the position of the covering hand, we identified three categories thanks to our manual analysis (the same three identified in [9]). The categories are the following: (i) "Over", when, from the point of view of the camera, the hand is mainly placed horizontally on the PIN pad; (ii) "Side", when the hand is shifted to the left side of the PoS and the rest of the fingers are spoon-bent to cover the typing hand; and (iii) "Front", when the fingertips are pointing to the top of the PIN pad (i.e., just about digits 1,2, and 3) and the hand bends to cover the PIN pad.

For the movement of the covering hand criterion, we identified three categories thanks to our manual analysis. To the best of our knowledge, this criterion has never been considered in previous works. The categories are the following: (i) "No movement", when the covering hand is touching the PoS or the table, remaining therefore immobile while typing thanks to the stable support; (ii) "Dependent", when the victim places the covering hand on top of the typing hand and moves with it while typing; (iii) "Independent", when the covering hand does not touch any support (nether the table, the PoS, or the typing hand). We classified these three categories as related to the movement of the hand since we noticed that the hand 1) is practically immobile when supported on the PoS ("No movement"); 2) it moves slowly when supported on the writing hand ("Dependent"); and 3) it moves quickly when not touching anything ("Independent").

In Table II, we report the Top-3 5-digit PIN accuracy and support (reported in parentheses) for our investigated criteria for all our datasets. The support corresponds to the number of participants available for a specific criterion and category. For both criteria, the reader may note that summing the reported supports the final number of participants is slightly lower than the number of collected participants reported in Section IV-B. This discrepancy is due to the participants who were removed from the test sets because they did not sufficiently cover the PIN pad (as explained in Section V-A).

As is visible from the table, we did not extrapolate the results for all six combinations of datasets/categories. This is due to the lack of appropriate support, which is too low to have a statistical meaning for some categories. This consideration is obvious for the 45° dataset where the support is polarized on one category for both position and movement ("Over" and "No movement", respectively). For the second dataset (iPP 220), it is good to remember that we are using a split participant-independent. Therefore, the data relating to a participant is always kept in the same set (training, validation, or test). Therefore, a single participant in the "Independent" movement style would result in too low a variance in the test set, thus distorting the results.

For the Position criterion, the table shows similar results for the first and the second data collection. The lowest accuracy is achieved by the covering style "Over". This category is the most effective since the exposed parts are reduced to a minimum. In particular, for the iPP 220 dataset, the reduction in accuracy for "Over" is particularly significant, with a decrease of 43% and 32% compared to "Side" and "Front",

TABLE II
ACCURACY RESULTS FOR THE INVESTIGATED TYPING STYLES. THE TABLE REPORTS THE RESULTS FOR EACH DATASET AND EACH CATEGORY. WE PROVIDE THE SUPPORT FOR EACH CATEGORY IN PARENTHESES.

Category / Dataset	Position			Movement		
	Over	Side	Front	No movement	Dependent	Independent
iPP 320	0.215 ± 0.170 (17)	0.241 ± 0.136 (24)	0.235 ± 0.221 (13)	0.277 ± 0.183 (26)	0.199 ± 0.167 (19)	0.202 ± 0.197 (9)
iPP 220	0.145 ± 0.101 (23)	0.252 ± 0.206 (4)	0.214 ± 0.228 (6)	/ (27)	/ (5)	/ (1)
iPP 320 45°	/ (38)	/ (0)	/ (0)	/ (38)	/ (0)	/ (0)

TABLE III
PERCENTAGE OF PARTICIPANTS FOR EACH COMBINATION OF CATEGORIES FOR THE DATASET IPP 320. IN PARENTHESES, WE PROVIDE THE ABSOLUTE NUMBER OF PARTICIPANTS.

Position / Movement	No movement	Dependent	Independent
Over	5.56% (3)	16.67% (9)	9.26% (5)
Side	22.22% (12)	14.81% (8)	7.41% (4)
Front	20.37% (11)	3.70% (2)	0% (0)

TABLE IV
PERCENTAGE OF PARTICIPANTS FOR EACH COMBINATION OF CATEGORIES FOR THE DATASET IPP 220. IN PARENTHESES, WE PROVIDE THE ABSOLUTE NUMBER OF PARTICIPANTS.

Position / Movement	No movement	Dependent	Independent
Over	57.58% (19)	9.09% (3)	3.03% (1)
Side	12.12% (4)	0% (0)	0% (0)
Front	12.12% (4)	6.06% (2)	0% (0)

respectively.

As explained above, the table reports the results for the movement criterion only for the first dataset (iPP 320). The category "No movement" achieved a higher accuracy of about 28%, showing that this covering style is less effective in protecting from our attack. The accuracy for both the Dependent and Independent styles is about 20%, but what is more interesting is the large variance for these two results (particularly for "Independent"). Based on the variance value, we can notice that participants who use these styles sometimes cover very well, while other times they cover badly (keeping on average an excellent covering quality). When analyzing the videos, we noticed (particularly for the participants of the "Independent" category) that participants move the covering hand, but in a way that is not always coordinated with the typing hand. This phenomenon could lead to greater visibility for some PINs and less visibility for others, giving rise to this large variance. Thus, these two covering styles can be very effective. Still, participants must make an effort to move the covering hand in a coordinated manner with the typing hand.

In Tables III and IV, we reported the support for all possible combinations of categories for PoS iPP320 and iPP220. The same table is not reported for PoS iPP320 $45°$ since all the samples would be in the combination "Over-No movement". The first thing we notice in both tables is that the "Front-Independent" combination has no support. This combination is quite unnatural since a participant would have to simultaneously (i) point the fingers to the top of the keypad, (ii) bend the hand to cover, and (iii) move quickly to cover the movements.

Generally, the support is much more evenly distributed for the first dataset. A fundamental difference concerns the "Over-No Movement" combination: while in the first dataset, this combination is used only 5.56% of the time, in the second, the value rises to 57.58%. This information is critical, as it suggests that the differences in size between the two PoS are significant to the point of influencing the coverage style chosen by participants.

C. Results for Different Distances of the Camera

The results we report above are related to an attack performed with a camera from a distance of 1 meter. In many cases, this condition can be satisfied without significant difficulty since the attacker can carefully select target PoSs for which placing the camera at 1 meter is possible. However, sometimes, the distance condition might be difficult to satisfy: the attacker might want to attack a specific PoS where placing a camera at 1 meter is impossible, or the camera might be a compromised surveillance camera pointing in the right position but at a higher distance.

In this section, we investigate the impact of the distance between the camera and the PoS on the performance of our attack. We investigated the results for our three configurations (i.e., PoS iPP 320, PoS iPP 220, and PoS iPP 320 $45°$). As explained in Sections IV-A and IV-D, the camera records frames at 1024p resolution, which are then cropped to a size of 260×300. In this experiment, we decreased the resolution of these input frames to simulate a camera at a higher distance. In particular, we know that, given the same area framed by a camera, the number of pixels decreases with the square of the distance. Therefore, at a distance of 2 meters, the resolution of our frame is reduced to a quarter (i.e., 130×150), at 4 meters, the resolution is an eighth (i.e., 65×75), and at 8 meters, the resolution is a sixteenth (i.e., 32×37). Figure 9 shows the results for the investigated distances. At a distance of 2 meters, the results are practically unchanged compared with the 1m distance. At a distance of 4 meters, the accuracy of our ML model begins to deteriorate, showing drops of up to 30% in accuracy. By doubling the distance again to 8 meters, we can see a significant drop in performance. In particular, for iPP320 and iPP220, the accuracy has already reached values below 10%, making it difficult to scale to greater distances. For iPP320 $45°$, the accuracy remains above 15%. We report the single key and the 4-digit PINs accuracy in the Appendix.

D. Study on All Possible Pairs of Keys

In the previous experiments, we investigated the results of our attack, focusing on the single key and N-digits PIN

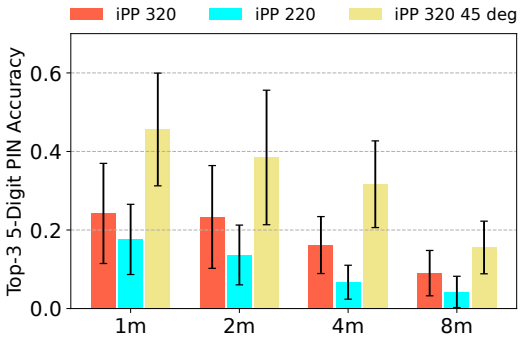


Fig. 9. Top-3 accuracy for 5-digit PIN reconstruction for different distances.

accuracies. In this section, we considered how the accuracy of a key varies concerning the key pressed immediately before. In this way, we ended up with pairs of keys, where for each pair, the first element is the previously pressed key, while the second is the target key whose accuracy we are interested in knowing. Therefore, when we write about a pair $(2, 5)$, we refer to the accuracy of a key 5 when 2 is the previous key within the PIN. Note that in this way, the first key of each PIN is never the second key of a pair, as there is no previous key within the PIN.

This experiment investigates whether there are pairs of keys whose accuracy significantly differs from the others. To achieve this aim, we used a T-test to compare a pair's average accuracy with all other pairs' average accuracy. A T-test is an inferential statistic used to determine if there is a significant difference between the means of two groups and how they are related. Since the accuracy for our three datasets is significantly different, a pair's accuracy is compared with the other pairs' average accuracy calculated on its dataset.

In Figures 10, we report the results of our experiments for all the pairs with 5 as the second key for our three datasets. Each figure contains 10 box plots showing the accuracy distribution for each pair. Therefore, the first columns report the box plots for the pairs $(0, 5)$, the second column the box plots for the pairs $(1, 5)$, and so on. The blue horizontal lines report the average accuracy value for all the other couples for the specific dataset.

Our experiment can give three results. First, the T-test shows no statistical difference between the pair's accuracy and the average accuracy. In this case, we colored the box plot red. Second, the T-test shows a statistical difference between the pair's accuracy and the average accuracy, and the pair's accuracy is higher than the average accuracy. In this case, we colored the box plot light blue. Third, the T-test shows a statistical difference between the pair's accuracy and the average accuracy, and the pair's accuracy is lower than the average accuracy. In this case, we colored the box plot green.

The interesting cases are the second and third, as both denote a pair with a significant difference from the global average. In particular, the box plots colored in light blue are related to a pair that is simpler than an average pair to guess. On the contrary, the box plots colored in green are related to a pair that is more difficult than the average pair to guess.

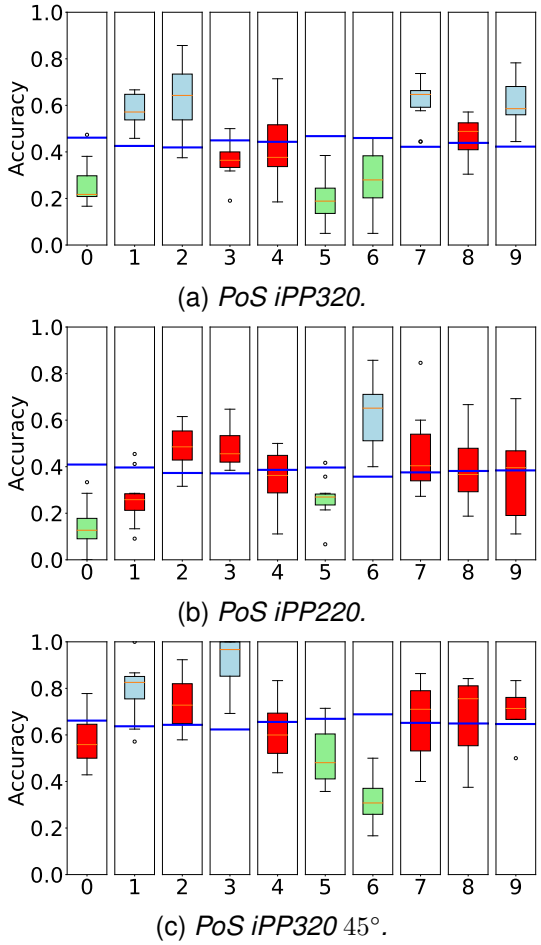
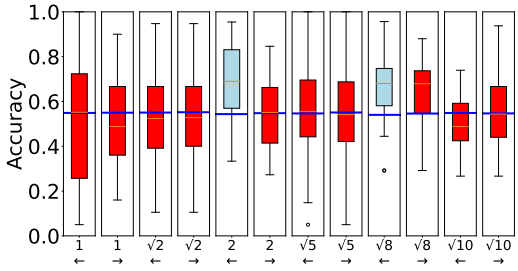


Fig. 10. The box plots report the results for the T-test. The graphics report the accuracies for all pairs of keys where the second key pressed is the digit 5 for our three datasets. We reported on the X-axis all 10 possible first keys of the pairs. The blue horizontal lines are the average for all the other couples in the data collection.

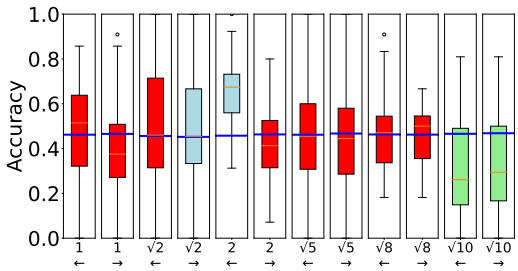
We calculated the T-test for all possible pairs of the dataset. We initially used a p-value threshold of 0.01, but then applied adjusted p-values to control for the false discovery rate, addressing the issue of multiple comparisons, which can increase the likelihood of false positives. From the results obtained, we notice that some pairs of keys for all three datasets obtained the same result from the T-test. Among the pairs that always showed a statistically meaningful higher accuracy compared to the average accuracy, we found $(9, 7)$. Instead, among the pairs that always showed a statistically meaningful lower accuracy compared to the average accuracy, we found $(5, 5)$. In general, these pairs help us as they can give a metric to define which PINs are more or less secure. We reported the box plots for other pairs and datasets in the Appendix.

E. Study on Keys Distances

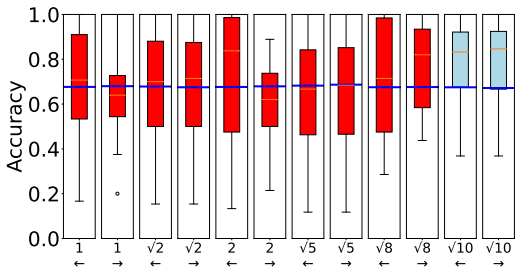
In this section, we investigate whether and how the distance between a target key and the key pressed immediately before it can affect the accuracy of the target key. To define the concept of distance, we first reused the concept of couple introduced previously. Specifically, given two keys, we define a Euclidean distance between the keys, where two keys have a



(a) *PoS iPP320.*

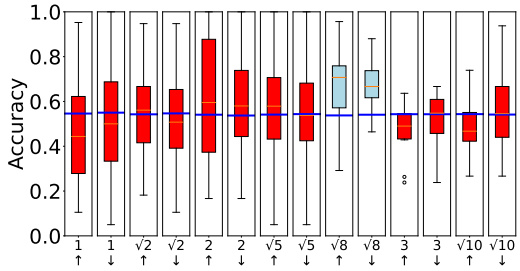


(b) *PoS iPP220.*

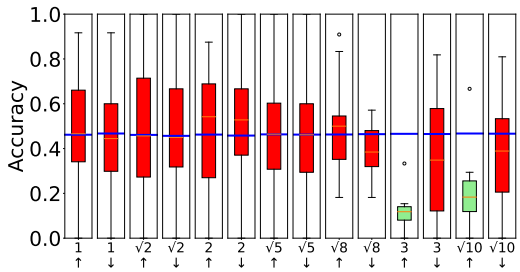


(c) *PoS iPP320* $45°$.

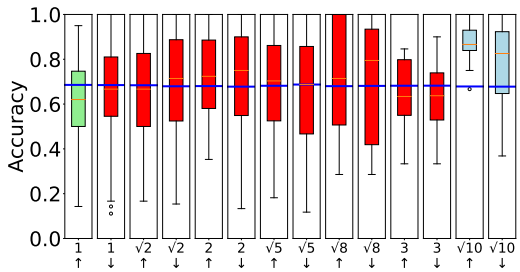
Fig. 11. Box plots for all the distances combined with the horizontal movements.



(a) *PoS iPP320.*



(b) *PoS iPP220.*



(c) *PoS iPP320* $45°$.

Fig. 12. Box plots for all the distances combined with the vertical movements.

distance of one if they are contiguous on the keypad vertically or horizontally (e.g., $(1, 2)$ or $(5, 8)$). It follows that, in our PIN pad, we can have integer distance values from 0 to 3, where 0 implies that the previous key was the same, while 1, 2, and 3 imply movement only vertically or only horizontally between the two keys. It follows that the distance of 3 will only be possible between keys 2 and 0, as no other pair of keys allows a distance of three vertical or horizontal. Then, there are pairs of keys for which the distance calculation involves the Pythagorean theorem. The resulting distances are $\sqrt{2}$ (e.g., $(4, 2)$), $\sqrt{5}$ (e.g., $(7, 2)$), $\sqrt{8}$ (e.g., $(7, 3)$), and $\sqrt{10}$ (e.g., $(1, 0)$).

Classifying pairs into distances is useful; however, further refinement is useful for extracting more detailed information. Indeed, if the pair (X, Y) has a distance N between its keys, the pair (Y, X) will also have the same distance. For this reason, we introduced the concept of direction of movement. In Figures 11 and 12, we combined the concept of distance with the concept of direction. In particular, in Figure 11, we reported all results for the distances that involve a horizontal movement (i.e., left or right), while in Figure 12, we reported the results for the distances that involve a vertical movement (i.e., up or down). We report the results using box plots as in Section V-D. We used the T-test with a p-value of 0.01 and adjusted it to avoid false positives. The colors maintain the same meaning as in the previous figures, and the blue lines still represent the average accuracy for all the other couples of pairs of the dataset. However, it is important to note that, in Figure 10 we refer only to the pairs where the second element is 5 (the other box plots are not reported for space limits), instead in Figures 11 and 12 a specific distance refers to all pairs which elements are represented by that distance.

We start comparing the results for PoS iPP320 and iPP220: these two dataset collections are both collected on PoS placed horizontally. Considering the results for the horizontal movements in Figures 11a and 11b, they have high results for the distances 2 left. This is interesting since we can notice that the opposite move (i.e., 2 right) is not statistically different from the average. This asymmetry may be due to the inherent asymmetry of typing with the right hand and covering with the left hand.

We can see an important difference between iPP 220 and iPP 320 $45°$. Indeed, the results for $\sqrt{10}$ are statistically lower compared to the average for iPP 220, while in iPP 320 $45°$, the same movement is statistically higher compared to its average. The only exception is $\sqrt{10}\downarrow$, which is still in line

with this trend even if it does not show significant statistical differences. This may be due to the different typing styles used by participants. In the iPP 220 dataset, the participants place their whole hands over the PIN Pad, and therefore, they can potentially use all their fingers to type. In iPP 320 $45°$, as previously reported, all participants grabbed the PoS with their hands and typed with their thumb. This difference in typing style may explain the contrasting results shown by the two datasets. Indeed, in iPP 220, we noticed that participants who are typing the numbers 1 and 3 tend to have their thumb already positioned near the number 0 and could, therefore, be incentivized to use this finger to press this button. The same also applies to the reverse condition for those participants who are pressing the number 0 and will have to press the number 1 or 3 (that will be pressed probably by the index). In this condition, the movement made by the hand is minimal, and the information that can be used by the deep learning model could, therefore, be insufficient to perform a good prediction. On the contrary, in PoS iPP 320 $45°$, participants always use their thumb to type. This implies that the participant must always move their finger on the target key. This consideration must be combined with the fact that $\sqrt{10}$ is the longest possible distance. Therefore, contrary to the PoS2 situation, in this case, the $\sqrt{10}$ move generates an accentuated movement of the hand, providing more significant features compared to the PoS2 dataset. A final consideration is that, for $\sqrt{10}$, iPP 320 does not have similar results to iPP 220. This could be due to the larger keypad size, which no longer causes the thumb to be close to 0 when the other fingers are close to 1 and 3.

F. Effect of PIN Popularity on Prediction Accuracy

To assess whether real-world PIN popularity affects attack performance, we designed an experiment to isolate the impact of PIN frequency on predictability. We hypothesized that the most common codes might be easier to guess because participants habitually type them more accurately or because simple patterns (e.g., 1111 or 1212) are inherently more predictable.

As ground truth for PIN frequencies, we employed the SecLists dataset [41], which aggregates multiple public sources and is proposed as a resource from Kali Linux [42]. Focusing on 4-digit PINs—the most extensively studied format—we sorted our test samples by descending frequency and partitioned them into ten equal-sized deciles (0–10%, 10–20%, …, 90–100%). We then ran our attack on each decile, measuring single-PIN accuracy. We run the experiment for all three datasets and all test sets of our 10-fold cross-validation.

The results reveal that accuracy fluctuates without any systematic trend across the ten frequency-based subsets. In other words, under our uniform pseudorandom PIN generation model, there is no observable correlation between a PIN's real-world popularity and its guessability.

G. Robustness to Varying Illumination Conditions

To evaluate the robustness of our attack under varying illumination conditions, we conducted a set of experiments

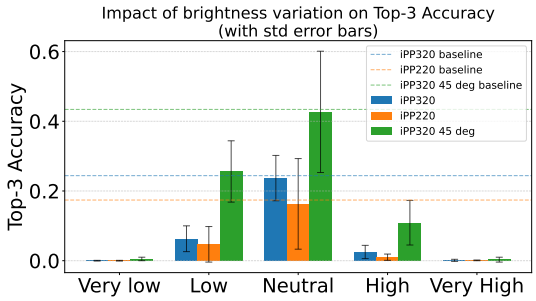


Fig. 13. Top-3 accuracy for different light conditions for all our datasets.

simulating different lighting scenarios. The goal is to assess how performance degrades or holds under realistic lighting fluctuations, such as those commonly found in physical retail environments. All experiments were performed on the test sets across all folds, applying synthetic brightness transformations to the input frames.

We defined five lighting conditions: very low, low, neutral, high, and very high. These represent increasingly extreme levels of brightness, with neutral approximating typical ambient indoor lighting. The neutral setting is used as a baseline to assess performance under minor, natural variations. Low and very low simulate dim or poorly lit environments, while high and very high correspond to direct sunlight or strong overexposure, such as camera flash.

To simulate these conditions, we applied controlled pixel intensity shifts to the video frames, uniformly modifying all RGB channels by adding or subtracting integer values. Specifically, intensity values were sampled uniformly at random within predefined ranges and then clamped to maintain pixel values in the valid [0, 255] range. The transformation ranges were: -150 to -120 (very low), -90 to -60 (low), -30 to +30 (neutral), +60 to +90 (high), and +120 to +150 (very high).

Figure 13 shows examples of the very low and very high conditions (subfigures a and b, respectively), alongside a bar plot summarizing the accuracy results for each condition across our three PoS scenarios. As expected, the highest accuracy is achieved in the neutral setting, which aligns closely with the original unmodified results, confirming that normal fluctuations in indoor lighting do not significantly affect prediction performance.

Interestingly, moderate increases in brightness (high) resulted in a more pronounced drop in accuracy than the corresponding decrease observed under low lighting. For low conditions, the system remained effective for all scenarios, while for high lux conditions, only iPP 320 45° still achieved good performances over 10%, while iPP 320 is a bit borderline with an accuracy of 3%. For the most extreme conditions (very low and very high), performance degraded substantially, rendering the attack ineffective.

These findings indicate that the proposed attack remains viable under typical real-world lighting conditions. The most extreme brightness shifts simulated in this experiment would be unlikely to occur in a retail environment, where both visibility and usability are crucial for customers. Therefore, the method demonstrates resilience in the presence of realistic

lighting variation.

VI. Discussion

A. Limitations

The effectiveness of the proposed attack heavily relies on the prediction capability of the model. Specifically, in the application of machine learning models, we assume that during the attack, the input data of the model will be drawn from the same probability distribution as the data contained in the training set (independent and identically distributed (IID) data). In our case study, we focused on two PoSes, which limited the diversity of our data collection. Therefore, testing the resulting model on videos recorded using a different dataset can only be accomplished by ensuring that the new data considered are IID with respect to our training set. Note that the attack can be carried out using two distinct strategies: (i) one focusing on a specific PoS, which involves utilizing a limited dataset that allows for consideration of a very specific probability distribution of the data; (ii) alternatively, one can opt for the more challenging scenario of targeting a model designed to accommodate a wide range of possible PoSes – this necessitates training the model using a vast dataset spanning numerous potential PoSes, thereby significantly increasing the complexity of data collection.

In our experiments, we partially considered the effect of the camera tilt. Indeed, the angle can be varied on two axes with respect to the keypad: right/left or forward/backward. The left/right angle was not considered in our experiments, as some experiments have been conducted by [9], showing the influence of an inclination up to 45°. Regarding the forward/backward angle, the possibilities are more limited. Tilting the camera backward (i.e., towards the bottom of the keypad) is difficult as the typing victim's head covers the keypad. About the forward inclination of the camera, we partially studied it in the experiment in which the PoS is inclined at 45°: this condition partly investigates a camera tilt forward. While our controlled setups confirm the feasibility of the attack, real-world camera placement remains a significant challenge. Capturing clear footage of finger movements requires precise positioning, which may be hindered by environmental constraints, existing surveillance, or the risk of user detection. Therefore, the practicality of the attack largely depends on the specific context and the attacker's access to the environment. Furthermore, our attack assumes a static keypad layout, which reflects the design of most traditional PoS systems. However, some advanced terminals or mobile applications might adopt randomized keypad layouts to prevent input pattern recognition. In such scenarios, our model would likely fail to generalize, as the finger movements would not correspond to fixed digit positions. This represents a limitation that should be considered when evaluating on which PoS to conduct the attack.

B. Security Implications and Defensive Insights

Our study demonstrates the potential of deep learning models to infer PINs from video recordings, even when the user attempts to conceal their input with their hand. This highlights a growing threat to traditional authentication methods and suggests that similar approaches could be adapted to attack gesture-based inputs or alphanumeric passwords.

An important implication of our findings is the possibility of combining the inference model with established PIN guessing strategies. For example, after obtaining a Top-N list of candidate PINs through visual analysis, an attacker could reorder them by leveraging statistical models of common PINs or user-specific priors (based on cultural habits or leaked datasets). In this context, works such as PointerGuess [43] show how a pointer-based mechanism can reuse knowledge of known passwords to effectively reorder hypotheses, while Fast, Lean, and Accurate [44] demonstrates that even lightweight neural networks can assign more accurate probabilities than traditional n-gram models. Similarly, Password Guessing Using Random Forest [45] provides an example of a machine learning model capable of classifying candidates based on learned patterns, and Targeted Online Password Guessing [46] highlights the advantage of integrating contextual information (such as previous passwords or demographic data) to reduce the number of required attempts.

Additionally, Understanding Human-Chosen PINs [40] shows that user-chosen PINs have an average entropy (8.41 bits for 4-digit PINs, 13.21 bits for 6-digit PINs) much lower than the theoretical maximum, thus offering further guidance to reorder hypotheses based on statistically likely sequences.

This can be an efficient technique in cases where a bank allows a user to reset their PIN.

In practical attack scenarios, an adversary could automate the entire pipeline: record the input with a hidden camera, process the video using our deep learning model, and run a PIN guessing strategy informed by public datasets or user-specific priors. The scalability of this approach is concerning, particularly in environments where surveillance devices can be covertly installed. In such cases, hundreds or thousands of users could be targeted with minimal human intervention.

Together, these factors demonstrate how deep learning models can become critical components of more complex and powerful multi-step attacks, posing a serious threat to the security of PIN-based systems.

Moreover, we observed that certain consecutive PIN pairs are inherently more predictable due to patterns in how users move their fingers or in the visual regularity of the motion. This introduces a subtle yet critical vulnerability: if some PIN sequences are easier to infer than others, even randomly generated PINs may offer varying levels of resistance to our attack. This insight opens the door to a novel defensive application of our results. Specifically, the model's ability to identify PINs that are more easily inferred could be leveraged in the design of more secure PIN generation systems. For example, a PIN could be selected not only to avoid common sequences (e.g., "1234"), but also to maximize visual unpredictability when entered on a keypad. Such an approach would enable the generation of PINs that remain robust even against advanced deep learning-based side-channel attacks.

Finally, the scalability of our attack, especially when coupled with automated video capture and PIN guessing, highlights the need for system-level countermeasures to reduce

exposure to this class of threats. One option is to shuffle the positions of digits on the keypad each time a PIN is entered, which prevents an attacker's model from learning fixed spatial patterns; however, forcing users to locate each number anew slows down entry and increases the likelihood of input errors, particularly for those with limited vision or dexterity. Another approach is to introduce subtle visual noise or dynamic overlays on the screen, such as changing refresh rates, flickering backgrounds, or randomized masking effects, to obscure exact finger movements from a hidden camera. At the same time, these techniques can effectively disrupt video-based inference, but they may also strain users' eyes, create a disorienting interface, or require hardware upgrades that not all PoS systems support. A third possibility involves deploying AI-based monitoring that flags suspicious behavior—like a phone camera aimed at the keypad or an unusual pattern of failed PIN attempts—and alerts staff or temporarily locks the terminal; this method does not alter the PIN entry process itself but carries overhead for real-time analysis, risks of false positives interrupting legitimate transactions, and potential customer discomfort at being observed. Finally, adopting multi-factor authentication—by requiring an additional verification step such as one-time passwords, biometric checks, or mobile push notifications—can greatly reduce reliance on the PIN alone and render video-based inference far less useful. However, this added layer of security may introduce extra cost, lengthen the checkout process, and create barriers for customers who lack compatible devices or are unfamiliar with the technology. In practice, choosing among these defenses (or combining them) demands careful balancing of enhanced security against slower transaction times, higher error rates, hardware and operational costs, and user acceptance.

C. Comparison with Prior Work

In Table V, we compare our attack with similar methods from the state-of-the-art. To ensure a fair comparison, we focused on approaches that operate under similar conditions, specifically those that use video data as input. In [9], the authors proposed an attack focused on ATM PIN Pads, where video from hidden cameras is used as input to a neural network. In [47], a thermal camera is used to infer the keys pressed by the victim. Finally, in [25], the authors exploit the inter-keystroke timing to infer the most probable sequence of typed digits. The results we obtained demonstrate a high level of effectiveness, particularly when compared with previous works that primarily evaluated their attacks on ATM PIN pads. Indeed, our results have been obtained on a PoS PIN Pad, which has dimensions much smaller compared to an ATM PIN pad (see Section IV-A for further details). Despite this disadvantage, in cases where the keypad is horizontal, our attack obtains results of the same order of magnitude as [9] and [47], while maintaining a Top-3 accuracy above 23%. Furthermore, our paper is the first to investigate the influence of the PIN pad tilt on the attack performance. In particular, in this specific circumstance, our attack achieves superior performance to all previous video input-based methods. Our paper extended the studies on different elements that can influence the efficiency of our attack. We extended the study of the covering styles that are better to defend against our attack and investigated the influence of the camera distance. Finally, we introduced analyses on the key pairs and hand movements that can most influence the results of our attack.

TABLE V
COMPARISON OF OUR ATTACK WITH OTHER ATTACKS ON PIN PADS BASED ON VIDEO INPUT.

	4-digit PINs Top-N Accuracy (%)		
	Top-1	Top-2	Top-3
Inter-keystroke Attack [25]	0.02	0.35	0.72
ATM PIN Attack [9]	29.61	37.06	41.12
Thermal Track attack [47]	15.54	27.79	33.63
Our attack iPP 320	18.10	26.71	31.40
Our attack iPP 220	13.32	19.94	23.27
Our attack iPP 320 45 deg	34.93	47.25	53.80

About thermal attacks, such as the one proposed in [47], infer PINs by detecting the residual heat left by fingers on the keypad. While these techniques can be effective, they require expensive thermal imaging equipment and must be executed immediately after the PIN entry, before the heat dissipates, typically within a few seconds. In contrast, our approach uses standard video input and does not rely on any physical traces left by the user. This makes it a passive attack, as it only requires a concealed camera to observe the scene without any interaction or specialized sensors. Furthermore, it is more scalable due to the high cost of thermal cameras. This flexibility makes the attack more practical in real-world surveillance scenarios.

In addition to these automated attacks, shoulder surfing represents a more traditional, manual form of PIN inference. While effective in some scenarios, shoulder surfing requires physical proximity, favorable line of sight, and direct observation during input, making it inherently limited in scalability and practicality. Moreover, it becomes largely ineffective when users shield the keypad with their hands. In contrast, our method remains viable even under such occlusion conditions, thanks to its ability to extract meaningful motion patterns from partially visible hand movements. This highlights the advantage of our approach in terms of both automation and robustness under realistic user behavior.

VII. CONCLUSION AND FUTURE WORK

This paper presents a novel deep learning-based attack on PINs on PoS machines. We conducted several months of user study to develop the attack and acquire enough examples to build a training set for our deep learning model. The experimental results are compelling, demonstrating a Top-3 accuracy of over 50% for 4-digit PINs and 45% for 5-digit PINs. These findings unequivocally illustrate the practicality of our approach in real-world scenarios. Our experiments were meticulously designed to account for various factors, including different PoS machines, their positions/angles, and the distances from the camera to the PoS machine. This comprehensive approach ensures the reliability and validity of our findings. Next, we analyze whether pairs of consecutive

digits make PINs easier to guess. This allows us to provide recommendations for stronger PINs.

Since our work demonstrated that attacks on PIN on PoS machines are realistic and powerful, future work should consider potential countermeasures. Unfortunately, a delicate balance between security and usability makes potential countermeasures more challenging to deploy. One option that sounds intuitive is to consider touchscreen PoS machines with randomized layouts, but it remains to be seen whether users can cover the screen with one hand and still efficiently type such PINs. Moreover, due to the widespread usage of PoS machines with physical keypads, such a countermeasure would take a lot of work (cheap) to deploy.

VIII. Acknowledgments

This work was supported by the European Research Council (ERC) under the Starting Grant ResolutioNet (ERC-StG-679158) and by the European Commission under the Horizon Europe Programme as part of the project SafeHorizon (Grant Agreement no. 101168562). The content of this article does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the authors.

APPENDIX

A. Keys Distances for 4-digit PINs

In Figure 14, we report the TOP-3 for 4-digit PINs accuracy for different distances of the spying camera. We observe a decreasing trend similar to that for 5-digit PINs.

In Figure 15, we show the TOP-1 single key accuracy for the three datasets for different distances. The results shown in this figure are interesting, as they show well how a small variation in accuracy in the single key classification can lead to a much more significant variation in Top-3 accuracy. This phenomena is visible by comparing Figure 15 with Figure 14.

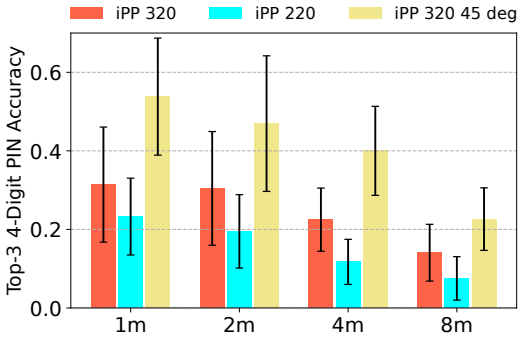


Fig. 14. Top-3 accuracy for 4-digit PIN reconstruction for different distances.

B. Other Significant Pairs of Keys

In this section, we report other box plots showing the statistical difference between the pairs' accuracy and the average accuracy. In particular, we show box plots showing the statistical difference for those pairs indicated as relevant. The pairs are (9, 7) for statistically higher accuracy and (5, 5) for statistically lower accuracy (the box plots for (5, 5) are already reported in the main paper).

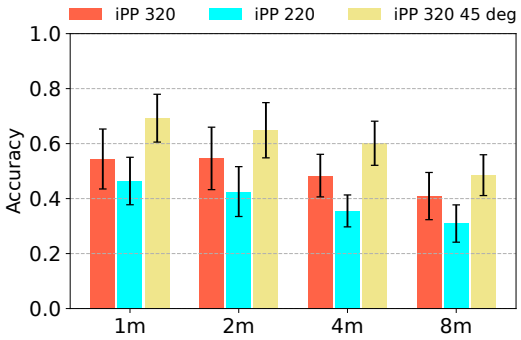


Fig. 15. Single key accuracy for different distances.

TABLE VI
Performance of our attack for the iPP320 dataset assuming different levels of frame detection error.

Frame error 3σ	Key accuracy	PIN TOP-3 accuracy
3	0.54	0.22
5	0.53	0.22
10	0.50	0.16
15	0.45	0.11

C. Study on Timestamp Precision

In this section, we investigate how the performance of our attack is affected by errors in the detection of the timestamps. In our attack, errors in timestamp detection can be traced back to a misalignment between the pressing of a key and the appearance of the asterisk on the PoS display. We conducted the experiments simulating various levels of timestamp errors. The experiment used the same training, validation, and test procedures as those used in the main experiments. The errors introduced were modeled using a normal distribution with a zero mean and different σ (standard deviation) values. Table VI reports the results of the conducted experiment. For convenience, the results are associated with a 3σ value. This metric is used as it indicates that most of the introduced errors on frame precision (specifically the 99.7%) fall within the reported 3σ range. The results show that the accuracy of our attack is the same up to a 3σ value of five. Above this value, the accuracy starts to decrease rapidly. This behavior aligns with the optimal window size of 11 frames that we found. Until a 3σ of five, most of the frames are inside our optimal window. Above a 3σ of five, the percentage of frames exceeding the window size becomes significant. In particular, the center of the window starts to be unrelated to the key press, and if buttons are pressed too fast, the collisions between the two buttons occur significantly more. In conclusion, the accuracy starts to decrease over this threshold. These results are important as a limit of five for the 3σ proves that a keystroke detection algorithm based on the screen asterisk recognition can be implemented due to the available margin of error.

TABLE VII
RESULTS FOR THE MALES AND THE FEMALES FOR THE THREE DATASETS.

	Male accuracy	**Female accuracy**
iPP 320	0.264 ± 0.138	0.195 ± 0.126
iPP 220	0.170 ± 0.122	0.218 ± 0.190
iPP 320 45°	0.373 ± 0.203	0.530 ± 0.247

D. Study on Gender-specific Performance

We report the model's accuracy for male and female subgroups across the three datasets collected in our study. As shown in Table VII, the results are split to highlight the performance differences between these subgroups. Despite the efforts to analyze the gender-specific performance, the differences observed are not statistically significant. The absence of statistically significant results indicates that the model performs consistently across genders within the scope of the datasets analyzed.

REFERENCES

[1] J. Allen, S. Carbo-Valverde, S. Chakravorti, F. Rodriguez-Fernandez, and O. P. Ardic, "Assessing Incentives to Increase Digital Payment Acceptance and Usage: A Machine Learning Approach," 2022.

[2] European Central Bank, "Study on the Payment Attitudes of Consumers in the Euro Area (SPACE) – 2022," https://www.ecb.europa.eu/stats/ecb_surveys/space/html/ecb.spacereport202212~783ffdf46e.en.html, 2022.

[3] Boston Consulting Group, "Global Payments Report 2023 ," 2023.

[4] M. Roland and J. Langer, "Cloning Credit Cards: A Combined Pre-play and Downgrade Attack on EMV Contactless," in *USENIX WOOT*, 2013.

[5] P. Kaur, K. Krishan, S. K. Sharma, and T. Kanchan, "ATM Card Cloning and Ethical Considerations," *Science and Engineering Ethics*, vol. 25, 2018.

[6] D. Basin, R. Sasse, and J. Toro-Pozo, "The EMV Standard: Break, Fix, Verify," in *IEEE Symposium on Security and Privacy*, 2021.

[7] ——, "Inducing Authentication Failures to Bypass Credit Card PINs," in *USENIX Security*, 2023.

[8] Kaspersky, "Tap-to-pay, Insert-to-Rob: Cybercriminals Can Now Block Contactless Payments," 2023. [Online]. Available: https://www.kaspersky.com/about/press-releases/2023_tap-to-pay-insert-to-rob-cybercriminals-can-now-block-contactless-payments

[9] M. Cardaioli, S. Cecconello, M. Conti, S. Milani, S. Picek, and E. Saraci, "Hand me your PIN! inferring ATM PINs of Users Typing with a Covered Hand," in *31st USENIX Security Symposium (USENIX Security 22)*, 2022, pp. 1687–1704.

[10] D. Shukla, R. Kumar, A. Serwadda, and V. V. Phoha, "Beware, Your Hands Reveal Your Secrets!" in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, 2014, pp. 904–917.

[11] G. Ye, Z. Tang, D. Fang, X. Chen, K. I. Kim, B. Taylor, and Z. 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.

[12] G. Pozzato, S. Michieletto, E. Menegatti, F. Dominio, G. Marin, S. Milani, and P. Zanuttigh, "Human-Robot Interaction with Depth-Based Gesture Recognition," in *Proc. of IAS-13 Workshop*, 01 2014.

[13] G. Boato, N. Conci, M. Daldoss, F. Natale, and N. Piotto, "Hand Tracking and Trajectory Analysis for Physical Rehabilitation," in *Proc. of IEEE MMSP 2009*, 11 2009, pp. 1 – 6.

[14] Ho-Joon Kim, J. S. Lee, and J. Park, "Dynamic Hand Gesture Recognition using a CNN Model with 3D Receptive Fields," in *2008 International Conference on Neural Networks and Signal Processing*, 2008, pp. 14–19.

[15] P. Molchanov, S. Gupta, K. Kim, and J. Kautz, "Hand Gesture Recognition with 3D Convolutional Neural Networks," in *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2015, pp. 1–7.

[16] S. S. Agaian, R. F. Pinto, C. D. B. Borges, A. M. A. Almeida, and I. C. Paula, "Static Hand Gesture Recognition Based on Convolutional Neural Networks," *Journal of Electrical and Computer Engineering*, Oct. 2019.

[17] N. L. Hakim, T. K. Shih, S. P. Kasthuri Arachchi, W. Aditya, Y.-C. Chen, and C.-Y. Lin, "Dynamic Hand Gesture Recognition Using 3DCNN and LSTM with FSM Context-Aware Model," *Sensors*, vol. 19, no. 24, 2019.

[18] P. C. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems," in *Annual International Cryptology Conference*. Springer, 1996, pp. 104–113.

[19] D. J. Bernstein, "Cache-Timing Attacks on AES," 2005. [Online]. Available: https://api.semanticscholar.org/CorpusID:2217245

[20] P. C. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," in *Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology*, ser. CRYPTO '99. London, UK, UK: Springer-Verlag, 1999, pp. 388–397. [Online]. Available: http://dl.acm.org/citation.cfm?id=646764.703989

[21] J.-J. Quisquater and D. Samyde, "ElectroMagnetic Analysis (EMA): Measures and Counter-measures for Smart Cards," in *Smart Card Programming and Security*, I. Attali and T. Jensen, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 200–210.

[22] D. Genkin, A. Shamir, and E. Tromer, "Acoustic Cryptanalysis," *Journal of Cryptology*, vol. 30, no. 2, p. 392–443, Feb. 2016. [Online]. Available: http://dx.doi.org/10.1007/s00145-015-9224-2

[23] M. Hutter and J.-M. Schmidt, "The Temperature Side Channel and Heating Fault Attacks," in *Smart Card Research and Advanced Applications: 12th International Conference, CARDIS 2013, Berlin, Germany, November 27-29, 2013. Revised Selected Papers*. Berlin, Heidelberg: Springer-Verlag, 2014, p. 219–235. [Online]. Available: https://doi.org/10.1007/978-3-319-08302-5_15

[24] D. X. Song, D. A. Wagner, and X. Tian, "Timing Analysis of Keystrokes and Timing Attacks on SSH," in *USENIX Security Symposium*, vol. 2001, 2001.

[25] K. Balagani, M. Cardaioli, M. Conti, P. Gasti, M. Georgiev, T. Gurtler, D. Lain, C. Miller, K. Molas, N. Samarin *et al.*, "PILOT: Password and PIN Information Leakage from Obfuscated Typing Videos," *Journal of Computer Security*, vol. 27, no. 4, pp. 405–425, 2019.

[26] X. Liu, Y. Li, R. H. Deng, B. Chang, and S. Li, "When Human Cognitive Modeling Meets PINs: User-Independent Inter-Keystroke Timing Attacks," *Computers & Security*, vol. 80, pp. 90–107, 2019.

[27] D. Balzarotti, M. Cova, and G. Vigna, "Clearshot: Eavesdropping on Keyboard Input from Video," in *2008 IEEE Symposium on Security and Privacy (sp 2008)*. IEEE, 2008, pp. 170–183.

[28] J. Sun, X. Jin, Y. Chen, J. Zhang, Y. Zhang, and R. Zhang, "VISIBLE: Video-Assisted Keystroke Inference from Tablet Backside Motion," in *NDSS*, 2016.

[29] J. R. Cazorla, J. M. De Fuentes, and L. González-Manzano, "Eye-based Keystroke Prediction for Natural Texts: A Feasibility Analysis," in *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2022, pp. 375–382.

[30] K. Mowery, S. Meiklejohn, and S. Savage, "Heat of the Moment: Characterizing the Efficacy of Thermal Camera-Based Attacks," in *Proceedings of the 5th USENIX conference on Offensive technologies*, 2011, pp. 6–6.

[31] T. Kaczmarek, E. Ozturk, and G. 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*, 2019, pp. 586–593.

[32] A. Compagno, M. Conti, D. Lain, and G. Tsudik, "Don't Skype & Type! Acoustic Eavesdropping in Voice-Over-IP," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ser. ASIA CCS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 703–715. [Online]. Available: https://doi.org/10.1145/3052973.3053005

[33] M. Cardaioli, M. Conti, K. Balagani, and P. Gasti, "Your PIN Sounds Good! Augmentation of PIN Guessing Strategies via Audio Leakage," in *European Symposium on Research in Computer Security*. Springer, 2020, pp. 720–735.

[34] W. Jin, S. Murali, H. Zhu, and M. Li, "Periscope: A Keystroke Inference Attack Using Human Coupled Electromagnetic Emanations," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 700–714. [Online]. Available: https://doi.org/10.1145/3460120.3484549

[35] J. Hu, H. Wang, T. Zheng, J. Hu, Z. Chen, H. Jiang, and J. Luo, "Password-Stealing without Hacking: Wi-Fi Enabled Practical Keystroke Eavesdropping," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 239–252. [Online]. Available: https://doi.org/10.1145/3576915.3623088

[36] T. Ni, X. Zhang, C. Zuo, J. Li, Z. Yan, W. Wang, W. Xu, X. Luo, and Q. Zhao, "Uncovering User Interactions on Smartphones via Contactless Wireless Charging Side Channels," in *2023 IEEE Symposium on Security and Privacy (SP)*, 2023, pp. 3399–3415.

[37] Statista, "Largest manufacturers of POS terminals worldwide from 2009 to 2021, ased on unit shipments," https://www.statista.com/statistics/373708/largest-global-pos-terminal-manufacturers-by-units-shipped/, 2023.

[38] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond Short Snippets: Deep Networks for Video Classification," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jun 2015, pp. 4694–4702. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CVPR.2015.7299101

[39] C. M. Bishop, "Training with Noise is Equivalent to Tikhonov Regularization," *Neural Computation*, vol. 7, no. 1, pp. 108–116, 1995.

[40] D. Wang, Q. Gu, X. Huang, and P. Wang, "Understanding Human-Chosen PINs: Characteristics, Distribution and Security," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, 2017, pp. 372–385.

[41] D. Miessler, "SecLists: Four-Digit PIN Codes Sorted by Frequency with Count," https://github.com/danielmiessler/SecLists/blob/master/Passwords/Common-Credentials/four-digit-pin-codes-sorted-by-frequency-withcount.csv, accessed: 2025-05-29.

[42] Kali Linux SecLists, "SecLists Package: Four-Digit PIN Codes Sorted by Frequency with Count," https://gitlab.com/kalilinux/packages/seclists/-/blob/kali/master/Passwords/Common-Credentials/four-digit-pin-codes-sorted-by-frequency-withcount.csv, accessed: 2025-05-29.

[43] K. Xiu and D. Wang, "PointerGuess: Targeted Password Guessing Model using Pointer Mechanism," in *33rd USENIX Security Symposium (USENIX Security 24)*, 2024, pp. 5555–5572.

[44] W. Melicher, B. Ur, S. M. Segreti, S. Komanduri, L. Bauer, N. Christin, and L. F. Cranor, "Fast, Lean, and Accurate: Modeling Password Guessability using Neural Networks," in *25th USENIX Security Symposium (USENIX Security 16)*, 2016, pp. 175–191.

[45] D. Wang, Y. Zou, Z. Zhang, and K. Xiu, "Password Guessing using Random Forest," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 965–982.

[46] D. Wang, Z. Zhang, P. Wang, J. Yan, and X. Huang, "Targeted Online Password Guessing: An Underestimated Threat," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 1242–1254.

[47] Y. Abdelrahman, M. Khamis, S. Schneegass, and F. Alt, "Stay cool! Understanding Thermal Attacks on Mobile-Based User Authentication," in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 2017, pp. 3751–3763.

Stefano Cecconello is a postdoctoral researcher at the University of Padua, specializing in cybersecurity and machine learning. He obtained his PhD in Brain, Mind, and Computer Science from the University of Padua, where he focused on the security of machine learning-based authentication systems. Before his current role, he was a postdoctoral researcher at TU Delft, working on 5G security and machine learning for network analysis. His research interests include developing advanced security solutions, particularly in machine learning and network security.

Matteo Cardaioli is an Innovation Manager at GFT Italy, where he leads the development of cutting-edge solutions in AI and digital transformation. In this role, he bridges academic research with practical applications, driving innovation in IT operations, cybersecurity, and AI-driven automation. With a background in cybersecurity research from the University of Padua, Cardaioli applies his expertise to enhance business processes and improve technological implementations. His work focuses on integrating advanced technologies like AI, machine learning, and multimodal systems into real-world enterprise environments.

Luca Pasa earned his Ph.D. in Mathematical Sciences (Computer Science) in March 2017. Since June 2024, he has been an Assistant Professor (RTDb) at the Department of Mathematics, University of Padova. His research focuses on Machine Learning, including Deep Learning, Computational Neuroscience, and Automatic Speech Recognition. Luca has contributed as a Program Committee member for multiple machine learning conferences and organized special sessions at international events. He is also an associate editor for IEEE Transactions on Neural Networks and Learning Systems (TNNLS) and a member of the IEEE Task Force on Deep Learning and the Italian Association for Artificial Intelligence (AIxIA).

Stjepan Picek is a professor at the University of Zagreb, Croatia, and an associate professor at Radboud University, The Netherlands. Before that, dr. Picek was an assistant professor at TU Delft, and a postdoctoral researcher at MIT, USA and KU Leuven, Belgium. His research interests are security/cryptography, machine learning, and evolutionary computation. Up to now, dr. Picek has given more than 50 invited talks and published more than 180 refereed papers. He is a program committee member and reviewer for a number of conferences and journals, as well as a member of several professional societies. Dr. Picek is a senior member of IEEE and an associate editor for several journals. He is a member of ELLIS and a Fellow of the Young Academy of Europe.

Georgios Smaragdakis (Senior Member, IEEE) is Professor of Cybersecurity with the Faculty of Electrical Engineering, Mathematics, and Computer Science, Delft University of Technology (TU Delft). He is also a researcher at the Max Planck Institute for Informatics and the Berlin Institute for the Foundations of Learning and Data (BIFOLD). In the past, he conducted research at MIT Computer Science and Artificial Intelligence Laboratory (CSAIL), MIT Internet Policy Research Initiative, TU Berlin, Boston University, and research labs (Akamai, Deutsche Telekom, Telefonica). His research was awarded a European Research Council Starting Grant, a Marie Curie International Outgoing Fellowship, Best Paper Awards at ACM SIGCOMM, ACM IMC, ACM CoNEXT, IEEE INFOCOM, IETF/IRTF Applied Networking Research Prizes, and selected for "Best of Computer Communication Review" and Communications of the ACM Research Highlights. He has been involved in the organization and technical program committees of many conferences, including, ACM CCS, ACM SIGCOMM, ACM IMC, ACM SIGMETRICS, ACM CoNEXT, ACM HotNets, ACM ASPLOS, ACM EuroSys, ACM Web Conference, IEEE S&P, IEEE EuroS&P, IEEE Infocom, IEEE HotWeb, USENIX ATC, USENIX NSDI, PETS, and ESORICS. He is an ACM Distinguished member.