VeriNet: User Verification on Smartwatches via Behavior Biometrics

Chris Xiaoxuan Lu
Department of Computer Science
University of Oxford

Bowen Du
Department of Computer Science
University of Warwick

Xuan Kan
College of Software Engineering
Tongji University

Hongkai Wen
Department of Computer Science
University of Warwick

Andrew Markham
Department of Computer Science
University of Oxford

Niki Trigoni
Department of Computer Science
University of Oxford

ABSTRACT

No longer reserved for nerdy geeks, nowadays smartwatches have gain their popularities rapidly, and become one of the most desirable gadgets that the general public would like to own. However, such popularity also introduces potential vulnerability. Until now, the de facto solution to protect smartwatches are passwords, i.e. either PINs or Android Pattern Locks (APLs). Unfortunately, those types of passwords are not robust against various forms of attacks, such as shoulder surfing or touch/motion based side channel attacks. In this paper, we propose a novel authentication approach for smartwatches, which adds another layer of security on top of the traditional passwords by considering the unique motion signatures when different users input passwords on their watches. It uses a deep recurrent neural networks to analyse the subtle motion signals of password input, and distinguish the legitimate users from malicious impostors. Following a privacy-preserving manner, our proposed approach does not require users to upload their passcodes for model training but only the motion data and identity labels.Extensive experiments on large-scale datasets collected real-world show that the proposed approach outperforms the state-of-the-art significantly, even in the most challenging case where a user has multiple distinct passcodes.

CCS CONCEPTS

•Security and privacy →Authentication; **•Human-centered computing** →Mobile devices;

KEYWORDS

Smartwatch, PINs, Motion Sensors

ACM Reference format:
Chris Xiaoxuan Lu, Bowen Du, Xuan Kan, Hongkai Wen, Andrew Markham, and Niki Trigoni. 2017. VeriNet: User Verification on Smartwatches via Behavior Biometrics. In *Proceedings of First ACM Workshop on Mobile Crowdsensing Systems and Applications, Delft, Netherlands, November 6–8, 2017 (CrowdSenSys),* 6 pages.
DOI: 10.1145/3139243.3139251

CrowdSenSys, Delft, Netherlands

DOI: 10.1145/3139243.3139251

1 INTRODUCTION

Arm and wrist-worn devices such as fitness trackers or smartwatches are seeing growing adoption. Many of these devices have screen displays for interaction, aka. smartwatches. It is estimated that 43 million smartwatches will be shipped in 2017, with sales doubling over the next four years to 86 million units in 2021. As they are currently designed as complementary accessories to smartphones, safeguarding on smartwatches have been overlooked. However, driven by the major players such as Google and Apple, smartwatches are expected to become more independent and able to revolve the mobile ecosystem. Even now, it is already to pay via a smartwatch without its paired phones [1]. Meanwhile, more private information are stored or pasted from smartphones to smartwacthes. Some recent apps also enable users to make personal photo galleries and save crucial health/fitness data on their smartwatches.

Featuring diversified apps and sensors, smartwatch greatly extends the functionality of traditional watch, but also appears to be a better option for attackers. PINs and Android Pattern Lock (APL) have been widely used in smartwatches for user authentication. However, recent studies found that attackers could leverage a bunch of side-channels to steal users' passcodes. For example, an attacker could leverage the oily residues left on the screen of a smart device to infer the pattern-like passcodes [2]. Motion sensors on smart devices are also found to leak passcodes, as screen taps of PINs results in changes of accelerometer and gyroscope readings. Although the above works mainly focus on smartphones, it is trivial to transfer these side channel attacks to smartwatches. Therefore, it is important to enhance the smartwatch's user authentication with a non-invasive user verification method, which is user-friendly and is able to further verify if the successfully logged-in user is the true owner of a smartwatch.

In this paper, we thrive to utilize user behaviors of entering passcodes for verification on smartwatches. The rationale behind our work is that watch owners have their own unique behavior patterns while entering their passcodes. Sharing the same flavor of other behavior-biometric based authentication, our proposed method can be leveraged to enhance the attacking resilience of smartwatches. In other words, we aim to design a complementary authentication tool of PIN- and APL-based systems that is able to deny the access of an attacker, even he accessed to the victim's passcode.

Although previous works have studied such behavior-based verification [8, 14, 20, 22], the problem we are going to solve in this work is more challenging in the following ways. First, the keys on

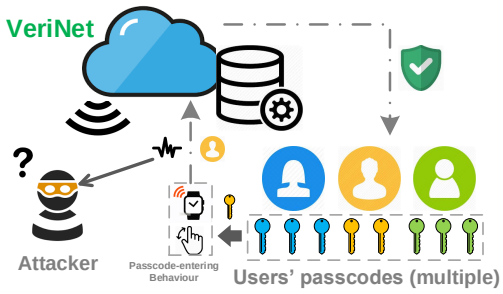

Figure 1: The work flow of proposed VeriNet, where the user authentication system is trained without the need of collecting users' passcodes but only passcode-entering motion data to the back-end cloud.

the much smaller smartwatch screen are very close to each other (compared to that on smartphone keypad or external keyboards), which makes identity inference difficult from motion data. Second, traditional methods need to access the users' passcodes for training the verification model. Not only this collection breaches users' privacy, but poses security threats when the passcode database leaks when back-end cloud is attacked.

Unlike previous works, we propose VeriNet, a deep recurrent neural network based user verification method that takes input as the motion data of passcode-entering behaviors. In order to demonstrate the effectiveness of our proposed approach, we collected more than 50k tuples of password-entering data (PIN and APL) and user identity, and successfully train VeriNet without passcode information. We show that VeriNet is able to handle subtle motions on smartwathes and preserve users' privacy (passcodes) to train a verification model.

2 BACKGROUND

In this section, we introduce the basics of the authentication systems on standard smartwatches. We then explain rationale behind the design of VeriNet.

2.1 Motion Induced by Passcode Input

Intuitively, entering both tapped and swiped passcodes will induce forces and orientation changes on the smartwatch. Since human skin has a certain level of elasticity, tapping on the smartwatch screen will cause minor displacement at the contact point along the vertical direction, i.e. the watch body will rotate for a small angle. Tapping causes an underdamped impulsive wave to develop, which causes small oscillations, shown in Fig. 2. On the other hand, when swiping passcodes, the pressing and friction force between the user's finger and touch screen will "drag" the smartwatch to move along both vertical and horizontal directions. This gives rise to small slip-pulse waves which have a longer duration than impulsive taps.

Recent studies show that the habits of entering passcodes vary a lot among people [7, 18], which can be used to further facilitate users. The rich variety of sensors on smartwatches make it possible to capture the differences of habits and we aim to capture them from motion. As shown in Fig. 2, as the user and impostor have different passcode-entering behavior, the induced motion data have different patterns.

In practice, induced motion of entering passcodes can be picked up by the Inertial Measurement Units (IMUs) embedded on most of the commercial smartwatches. IMU sensors have been widely used in many mobile sensing scenarios, since they are able to capture displacement and rotation of the devices in 3-D space, and become increasingly cheap and power efficient. Concretely in this work we consider both accelerometers and gyroscopes, which capture the linear acceleration and angular velocity (roll, pitch and yaw) with respect to the three axis.

2.2 Key Challenges

Sensor Noises: The idea of using motion data to decision-support authentication has been exploited on smartphones. However, verifying a user from motion data harvested on smartwatches is far more challenging than that on smartphones. Fig. 2 shows examples where the motion sensor readings change as the user taps a PIN or swipes an Android Pattern Lock (APL) on the smartwatch. As we can see, since smartwatches are physically much smaller than smartphones, the motion induced by passcode entries can be tiny, especially for APLs. Fig. 2 also confirms this with the signal-to-noise (SNR) ratio of motion sensors on different devices. We see that motion signals on smartwatches are far noisier, and can be 20-40dB worse than that of smartphones or high-end IMUs. In the presence of such low SNR, existing techniques designed for smartphones [17, 22] typically fail to work, since they use hand engineered features given the much weaker and noisier motion signals on smartwatches.

Privacy v.s. Accuracy: Developing a behavior-oriented verification system is essentially learning a classifier which takes sensor readings as inputs and returns user identifications. To this end, traditional methods [8, 14] collect both user's passcode and their respective motion samples on the client side and then uploads them to the cloud for classifier training. However, sending user's passcodes to the cloud already raises privacy concerns. This problem gets severer when the users have multiple passcodes to maintain, no prior art is able to cope with this scenario. A privacy-preserved mechanism [12, 15] should be considered to protect users' passcodes. By contrast, as shown in Fig. 1, VeriNet is free of asking users for their passcodes but only motion data.

3 DEEPLY LEARNED BEHAVIOR VERIFICATION

Unlike existing work, VeriNet considers a novel deep learning based authentication system, which does not rely on hand-crafted features, and is able to verify multiple passcodes of the same user.

3.1 Behavior Verification via Classification

As in [17, 22], we consider the task of user authentication as a *classification* problem, where category labels are a set of users. We expect the users are crowd-sourced participants. Then given the motion data captured by IMU sensors (accelerometers and gyroscopes), the problem of user authentication that the user has just input becomes that of finding a label within the identity database I, which can best explain the observed motion data. Notably, unlike

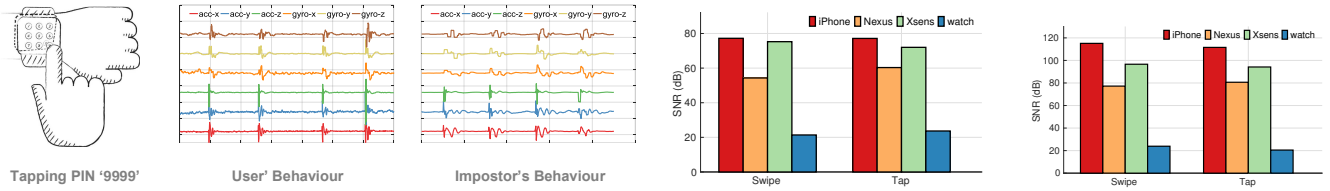

Figure 2: Left: examples of motion sensor data changes induced by tapping the same PIN but different people. The dashed watchface frames are the positions after or before a tap. Tapping a PIN on watch screen induces several impulse waves and gaps in between, and shows different entering habits. Right: SNR of motions sensors on different mobile ends.

previous works, VeriNet does not train different classifiers for every possible passcode, since it is unrealistic due to the searching spaces (389, 112 APLs and 10,000 PINs) and raises privacy concerns when asking users for their passcodes. Instead, our proposed method can take input as any motion samples of various passcodes, as long as it comes from the same user, i.e., same label. The rationale behind this we expect to design a classifier that accurate enough to differentiate users and impostors, and powerful enough to recognize various patterns of different passcodes of the same user.

3.2 Recurrent Neural Networks (RNNs) based Sequence Learning

The goal of Verification is associating the passcode-entering behavior to the user identity in database I. As discussed in Sec. 2, behavior differences among users can be reflected by the motion data captured by IMU sensors, when entering the passcodes on the watch. Compared with existing works on sequence modeling, our context is more difficult in the following ways: a) the motion data has variable size; and b) the temporal decencies in motion data implies identity information, e.g. the transitions from one digit to another contain vital content about user habits (see Fig. 2). Therefore, in this paper we use RNNs to model the motion data [21], which can take arbitrary length of input, and return the most likely identity label as output. We started from explaining how the RNN architecture work in our context.

Basic RNN Architecture: At each timestamp $\mathbf{t}$ a standard RNN maintains its internal hidden state $\mathbf{h}_t$ to interpret the temporal correlations, and given an input $\mathbf{x}_t$, the RNN updates its state by:

$$\begin{aligned} \mathbf{h}_t &= \mathcal{H}(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{b}_h) \\ \mathbf{u}_t &= \mathbf{W}_{hu}\mathbf{h}_t + \mathbf{b}_u \end{aligned} \tag{1}$$

where $\mathbf{W}_{xh}$, $\mathbf{W}_{hh}$ are the weights of the current input $\mathbf{x}_t$ and previous state $\mathbf{h}_{t-1}$, and $\mathbf{b}_h$ is the bias vector. $\mathcal{H}$ is an element-wise non-linear activation function, e.g., sigmoid or hyperbolic tangent function. The network output $\mathbf{u}_t$ is evaluated as a linear combination of the updated hidden state $\mathbf{h}_t$ and a bias vector $\mathbf{b}_u$.

Note that, $\mathbf{u}_t$ may appear at multiple timestamp, or only exist at a single (in most cases the last) timestamp. As discussed above, the problem studied in this paper is to compute the most likely user (i.e. label) within the database I given the sequence of motion data. We hence only focus on those RNNs with a single output (at last timestep). Fig.3 (Left) illustrate the work flow of standard RNN.

Although in theory RNN is able to model sequences with arbitrary length, in practice it often suffers from the gradient vanishing and exploding problems [10]. That is, it cannot capture the long-term dependencies well when the length of input sequences becomes large. As in our case the length of input sequences are often of hundreds of samples, we adopted Long Short Term Memory (LSTMs) [11] units in our RNNs, which is proved to be effective in preserving long-term dependencies.

Bi-directional RNN (BRNN): Standard RNN forwardly propagates information from the header of the sequence to the tail. LSTM helps to solve the long-term dependencies problem, but the beginning information is hard to pass through to the end due to the nature of any forward recurrent network. This issue gets severer when the input sequences are relatively long. However, as we can see in Fig. 4, the duration of entering passcodes are very long compared to normal study cases of RNN. Some APLs can last for above 5 seconds which leads long motion sequences with the size above 1,000 data points (sampling rate at 200Hz). The output at the last timestep contains little information about the beginning part of the data. Unfortunately, as pointed out in [8, 9], the header motion data of entering passcodes conveys certain information of user habits (from tapping the first digit to the second digit), losing it will degrade the overall classification performance.

To address this, VeriNet proposes a Bidirectional RNN (B-RNN) to capture the rich temporal dependencies within the input motion data. Unlike standard RNN, at each timestamp k the proposed B-RNN keeps two hidden states $\overleftarrow{\mathbf{h}}_t$ and $\overrightarrow{\mathbf{h}}_t$, which incorporate the future $(t+1, ..., T)$ and past $(1, ..., t-1)$ information in the input sequence respectively, as shown in Fig. 3 (Right). Then B-RNN uses the same strategy as in RNN to update those states from both directions. However, there are two output nodes in the network: one $\overrightarrow{\mathbf{h}}_T$ at the end and the other $\overleftarrow{\mathbf{h}}_1$ at the beginning. Therefore B-RNN maintains information flows from both the start and the end of the input sequence, and the output of the network is generated from the concatenation of the two output variables $\overrightarrow{\mathbf{h}}_T$ and $\overleftarrow{\mathbf{h}}_1$. As shown in the next section, the B-RNN based VeriNet is able to preserve the long-term dependencies in the motion signals, and achieves better accuracy than standard RNN based one.

3.3 Post-process Probabilistic Predictions

A limitation of above classification framework is that it only infers identities in the database. Unfortunately, in real world, the motion samples and identities of impostors are hardly to obtain. When the crowd-sourced identity database is large enough, this issue will

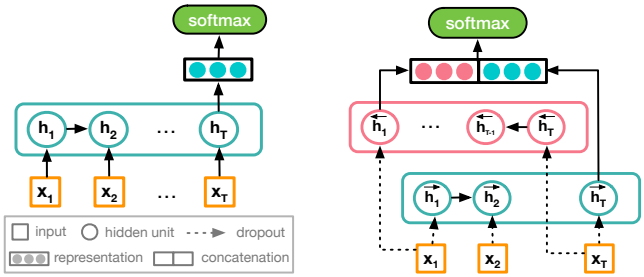

Figure 3: The architecture of Deep RNNs. Left: basic RNN. Right: B-RNN.

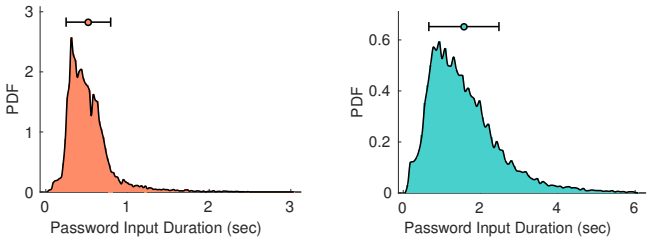

Figure 4: PDF of passcode input duration. Left: PIN. Right: APL.

become alleviated as impostors' passcode-entering behavior will somehow be similar to someone in the database. As long as the 'most similar' person is not the device owner, VeriNet still works well in such scenario for its high in-database classification accuracy. However, to be more robust to impostors, VeriNet adopts a post-process procedure. Fortunately, the softmax layer of BRNN returns the probability (likelihood) of all the identities, which makes it possible to filter out invaders outside the database. Instead of using the hard labels by the network, VeriNet firstly rejects guests when their predictions are far from everyone in the database. In other words, the probabilities computed by the softmax layer tends to be flat, and confirms they are outside impostors. We empirically set this threshold of flatness to be 0.7 for the max probability in the predictions.

4 EVALUATION

We evaluate the proposed VeriNet extensively on large-scale real world datasets collected in three different sites: *Oxford*, *Shanghai* and *Harbin*. The study involved 310 participants and accumulated over 60k passcode entries, with users wearing the smartwatch on the left hand[1].

4.1 Experiment Setup

Participants Enrollment: As discussed in Sec. 3, VeriNet verifies user identities by classifying the motion data of passcode-entering behavior into one of identities in the database I. 265 students from *Oxford*, *Shanghai* and *Harbin* participated as the user cohort. 155 of them are APL users and the other 110 are PIN users. On the other hand, we also recruited an impostor cohort, which consists of 45 students (in total 310 participants). The user cohort contributes their motion data of entering different passcodes on the smartwacth (Sony SW3 and iWacth) and we use these data for training identity inference model. We then ask the impostor cohort to reproduce the same passcodes on users' watches and use their samples for testing.

Passcode Surveying: To justify the experiment design and following passcode selection rule in [22], the involved passcodes in the evaluation are popular passcodes. However, we can *not* ask above cohorts to provide their own passcodes as it breaks our ethical agreement and there are no off-the-shelf database for use. Therefore, we surveyed anonymous participants for their personal passcodes (both PIN and APL) through a web app. We have made sure that this survey complied with data privacy policies, and there is no link between the collected data and any individual participant. We have first obtained the participants' consent that their data will be used in a scientific study to evaluate password security on smartwatches. If a participant agreed to proceed, she was then given a link to the website, where we asked her to contribute her passcodes via a designed web app. Through this way, we avoid any link between passcodes and cohorts who entering passcodes on the smartwach as well. After the survey, we fortunately obtained 112 PINs and the same amount of APLs. Among them we have 79 distinct PINs and 64 distinct APLs. Note that, we do *not* need passcode labels for training VeriNet, we do this survey step to ensure our that tested passcodes not favor us but follows the real-world passcode distribution.

Passcode-entering Data Collection: Each participant of user and impostor cohort was randomly given 6 APLs and 6 PINs from our surveyed passcode pool. This step is to simulate the case in real world that a user may have multiple passcodes. The participants were asked to wear the smartwaches on their left wrist but in the most comfortable way, and then enter each password in our data collection app about 20 times. To avoid overfiting the motion samples, we split the 20 times in several days (5 days on average per user) to make the data from the same user contains certain diversity. The app logs the ground truth by monitoring tap/swipe on the smartwatch screen, and saves the motion data at the same time. In total, we have collected 36,569 valid samples, each of which contains an user identity and the motion data when it was entered. Note that, VeriNet can be trained without passcodes but only identity labels, therefore we do not collect the passcodes user entered.

Competing Approaches: We implement the deep RNNs considered in VeriNet using Keras [5] with Theano [3] backend, and train them on NVIDIA K80 GPUs with the Adam Optimiser [13]. To the best of our knowledge, VeriNet is the first work to study the problem of user verification on smartwatches. Therefore, we compare VeriNet with the state-of-the-art typing behavior approaches, which are originally designed for smartphones. We evaluated two most known passcode-entering behavior-based approaches: `ICNP14` [22] and `Mobicom13` [17], both of them use handcrafted features and train specific classifier for every passcode. `ICNP14` extracts 4 sets of features including acceleration of tapping, touch pressure, touched size and password-entering time. It then uses an one-class learning model to reject impostors as long as the 'distance' between testing

[1]The study has received ethical approval *R50768*.

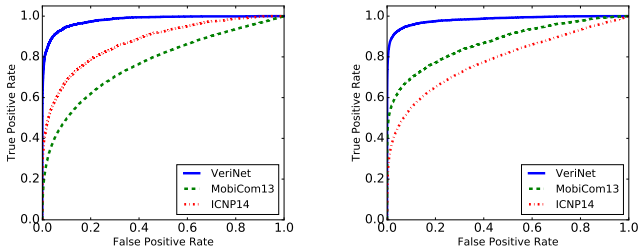

Figure 5: Average ROC curves of VeriNet and competing approaches. Left: PIN. Right: APL.

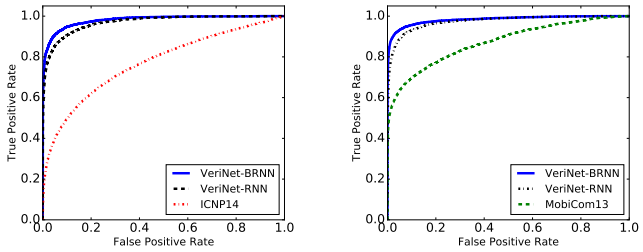

Figure 6: Impact of network architecture in VeriNet. Left: PIN. Right: APL.

samples and user templates are larger than a pre-difined threshold. `Mobicom13`, on the other hand, extracted 7 sets of features of two categories. The *stoke based category* contains inter-stroke time, stroke displacement magnitude and stroke displacement direction. The *sub-stroke based category* includes velocity magnitude, velocity direction and device acceleration. A SVM classifier is trained to determine the users or impostors based on the extraction features.

Evaluation Metrics: Following previous works [8, 14, 17, 22], we introduce the concepts of false positive rate (FPR), true positive rate (TPR), equal error rate (EER) as our evaluation metrics and and receiver operating characteristic (ROC). EER and ROC are defined as follows:

- EER: a biometric security system algorithm used to predetermines the threshold values for its FPR and TPR. When the rates are equal, the common value is referred to as the equal error rate. The value indicates that the proportion of false acceptances is equal to the proportion of false rejections.
- ROC: A graphical plot that visualizes the performance of a binary classifier as its discrimination threshold varies. ROC is created by plotting the fraction of the true positive rate (i.e., rejection rate when the user is invalid) vs the false positive rate (i.e., rejection rate when the user is valid), at various threshold setting ROC is a more complicated indicator, which reflects the performance of a system under different settings.

Although our problem formulation given in Sec. 3.1 is essentially a multi-class classifier, its predictions can be trivially transferred to binary verification results for user/impostor identification. The ROC is computed with the transferred binary results.

Passcode	Method	FPR(%)	TPR(%)	EER(%)
PIN	VeriNet	10.24	79.23	**7.17**
	MobiCom13	24.60	52.96	29.76
	ICNP14	19.44	63.95	20.64
APL	VeriNet	19.91	85.79	**6.09**
	MobiCom13	26.63	71.16	21.63
	ICNP14	24.00	54.49	28.67

Table 1: Performance comparison between VeriNet and competing approaches

4.2 Results

Overall Performance: This experiment evaluates the performance of identity verification given the passcode-entering motion data. As shown in Fig. 5, VeriNet completely outperforms competing approaches in terms of ROC on both PIN and APL datasets. For PINs, the area Area Under Curve (AUC) of VeriNet is nearly 20% better than the best competing approach `MobiCom13`. The performance gap stays significant when it comes to APLs, though the best competing approach changes to `ICNP14`. This is because `MobiCom13` is originally designed for swipe-based passcodes like APLs, while `ICNP14` focuses on the tap-based passcodes. More specially, as shown in Tab. 1, VeriNet is able to achieve 3 to 4 fold better performances in terms of all metrics. Its average EER for PINs is 7.17% and even lower for APLs with an average EER of 6.09%. By contrast, the average EER of both competing approaches are above 20%, which makes them impractical as a complementary authentication mechanism on smartwatches.

Impact of Network Architecture: As discussed in Sec. 3.2, we adopted the bi-directional RNN (BRNN) architecture over RNN in VeriNet. Fig. 6 shows the impacts of different network architectures. The AUC of BRNN-based model is nearly 7% superior to the RNN-based one for both PIN and APL cases. But as we can see, even the RNN-based VeriNet is still significantly superior to the best competing approaches in two cases. More specially, the EER improvement is $\sim$ 2% on both PIN and APL datasets, which implies that both ends of the passcode-entering data reveal identity information and should be considered for verification.

Moreover, we also observed that the regardless what network architecture is used, VeriNet has worse FAR but better TPR on APL dataset, compared to the results on the PIN dataset. It implies that, the behavior of tapping PINs varies more than swiping APLs for the same user; however, the behavior of swiping APLs is not discrimination enough among people as the one of tapping PINs.

Passcode	Method	FPR(%)	TPR(%)	EER(%)
PIN	BRNN	10.24	79.23	**7.17**
	RNN	11.72	74.71	9.72
APL	BRNN	19.91	85.79	**6.09**
	RNN	19.88	79.78	8.03

Table 2: Impact of network architecture on verification performance

5 RELATED WORK

5.1 Inferring Information from Motion Sensors

Researchers have attempted to infer keystrokes on smart wearables via motion sensors [4, 16, 19]. The core idea behind these works is similar: keystrokes on device screen lead to distinct force/attitude patterns. The motion data on smart wearables can thus be used to infer entered secrets. TouchLogger [4] and ACCessory [16] are early works, where ACCessory uses accelerometer only and TouchLogger utilizes both accelerometer and gyroscope to infer PINs. Similarly, TapLogger [19] refines previous techniques and uses a gyroscope to predict PIN-like secrets on smartphones. TapLogger uses a k-means clustering approach to extract the most likely classes (typically top 3). Given substantial observations of the secret (e.g. 32 PIN entry events), this is sufficient to estimate the true secret.

VeriNet obviously differs from them as we use motion data to harvest users' habits of password entering rather than inferring the input secrets.

5.2 User Authentication via Keystroke Operations

Using keystroke operations to authenticate users is relatively new topic that has yet to capture extensive research attentions. Several recent work has studied how to improve the touch unlocking mechanism by considering touch biometrics. Such work includes [6, 8, 22]. De Luca et al. in [6] propose to track touch data of slide operations to unlock the screen. Touch data including time, position, size and pressure are used directly to authenticate users. Frank et al. conducted a study on touch input analysis for smartphone user authentication, which is referred to as touch biometrics [8]. Based on a set of 30 behavioral features, the authors built proof-of-concept classifiers that can pinpoint a target user among 41 users with very low equal error rate. Shahzad et al. discuss a slide-based user authentication scheme, where a series of customized slides are used jointly to authenticate users [17]. The extracted features works well for swipe-based passcodes, e.g., APLs. By contrast, Zheng et al. designs the feature of keystrokes that is suitable to the tap-based passcodes, e.g., PINs. Moreover, it is able to train a verification model without the data from the impostors' side, which fits in the real world well. Yang et. at. further improve previous works that harvest multiple sensor modalities to classify finger tap events on smartphones. However, all prior arts needs the user's passcode and become cumbersome when a user has multiple passcodes.

Unlike above work, VeriNet is the first verification work based on passcode-entering behaviors that designed for smartwatches. Not only the SNR is much lower for the smartwatch case, VeriNet avoids collecting users' passcodes for classifier training and the learnt classifier is powerful enough to accommodate authentication for multiple passcodes from the same user.

6 CONCLUSION

In this paper, we propose a novel authentication approach for smartwatches, which adds another layer of security on top of the traditional passwords by considering the unique motion signatures when different users input passwords on their watches. It uses a deep recurrent neural networks to analyse the subtle motion signals of password input, and distinguish the legitimate users from malicious impostors. Unlike prior art, VeriNet is free of collecting users' passcodes for classifier training and the learnt classifier is powerful enough to accommodate authentication for multiple passcodes from the same user. Having collected tested on 310 participants, VeriNet is demonstrated to be 3-4 fold better than competing approaches.

REFERENCES

[1] Ltd Alipay.com Co. 2017. Alipay - Makes Life Easy. https://itunes.apple.com/us/app/alipay-makes-life-easy/id333206289?mt=8. (2017).

[2] Adam J Aviv, Katherine L Gibson, Evan Mossop, Matt Blaze, and Jonathan M Smith. 2010. Smudge Attacks on Smartphone Touch Screens. *USENIX Workshop on Offensive Technologies, Woot* (2010).

[3] James Bergstra, Frédéric Bastien, Olivier Breuleux, Pascal Lamblin, Razvan Pascanu, Olivier Delalleau, Guillaume Desjardins, David Warde-Farley, Ian Goodfellow, Arnaud Bergeron, and others. 2011. Theano: Deep learning on gpus with python. In *NIPS 2011, BigLearning Workshop, Granada, Spain*, Vol. 3.

[4] Liang Cai and Hao Chen. 2011. TouchLogger: Inferring Keystrokes on Touch Screen from Smartphone Motion.. In *Proceedings of the 6th USENIX Conference on Hot Topics in Security, HotSec*. USENIX.

[5] François Chollet and others. 2015. Keras: Deep learning library for theano and tensorflow. *URL: https://keras. io/k* (2015).

[6] Alexander De Luca, Alina Hang, Frederik Brudy, Christian Lindner, and Heinrich Hussmann. 2012. Touch me once and i know it's you!: implicit authentication based on touch screen patterns. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, (CHI)*.

[7] Dinei Florêncio, Cormac Herley, and Paul C Van Oorschot. 2014. passcode Portfolios and the Finite-Effort User: Sustainably Managing Large Numbers of Accounts.. In *USENIX Security Symposium*.

[8] Mario Frank, Ralf Biedert, Eugene Ma, Ivan Martinovic, and Dawn Song. 2013. Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. *IEEE transactions on information forensics and security* (2013).

[9] Marian Harbach, Alexander De Luca, and Serge Egelman. 2016. The anatomy of smartphone unlocking: A field study of android lock screens. In *ACM Conference on Human Factors in Computing Systems, CHI*.

[10] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, and Jürgen Schmidhuber. 2001. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. (2001).

[11] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. 9, 8 (1997), 1735–1780.

[12] Haiming Jin, Lu Su, Houping Xiao, and Klara Nahrstedt. 2016. INCEPTION: incentivizing privacy-preserving data aggregation for mobile crowd sensing systems.. In *MobiHoc*.

[13] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *International Conference on Learning Representations, ICLR*.

[14] Lingjun Li, Xinxin Zhao, and Guoliang Xue. 2013. Unobservable Re-authentication for Smartphones.. In *NDSS*.

[15] Chenglin Miao, Wenjun Jiang, Lu Su, Yaliang Li, Suxin Guo, Zhan Qin, Houping Xiao, Jing Gao, and Kui Ren. 2015. Cloud-enabled privacy-preserving truth discovery in crowd sensing systems. In *SenSys*.

[16] Emmanuel Owusu, Jun Han, Sauvik Das, Adrian Perrig, and Joy Zhang. 2012. ACCessory: passcode inference using accelerometers on smartphones. In *HotMobile*.

[17] Muhammad Shahzad, Alex X Liu, and Arjmand Samuel. 2013. Secure unlocking of mobile touch screen devices by simple gestures: you can see it but you can not do it. In *Proceedings of the 19th annual international conference on Mobile computing & networking, (MobiCom)*.

[18] Elizabeth Stobert and Robert Biddle. 2014. The passcode life cycle: user behaviour in managing passcodes. In *USENIX Symposium On Usable Privacy and Security*.

[19] Zhi Xu, Kun Bai, and Sencun Zhu. 2012. Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors. In *Proceedings of the 5th ACM conference on Security and Privacy in Wireless and Mobile Networks*.

[20] Hongji Yang, Lin Chen, Kaigui Bian, Yang Tian, Fan Ye, Wei Yan, Tong Zhao, and Xiaoming Li. 2015. TapLock: Exploit finger tap events for enhancing attack resilience of smartphone passcodes. In *IEEE International Conference on Communications, (ICC)*.

[21] Shuochao Yao, Shaohan Hu, Yiran Zhao, Aston Zhang, and Tarek Abdelzaher. 2017. Deepsense: A unified deep learning framework for time-series mobile sensing data processing. In *International Conference on World Wide Web, WWW*.

[22] Nan Zheng, Kun Bai, Hai Huang, and Haining Wang. 2014. You are how you touch: User verification on smartphones via tapping behaviors. In *IEEE 22nd International Conference on Network Protocols (ICNP)*.