DeepAuth: In-situ Authentication for Smartwatches via Deeply Learned Behavioural Biometrics

Chris Xiaoxuan Lu[†], Bowen Du[§], Peijun Zhao[†], Hongkai Wen [*§], Yiran Shen[¶], Andrew Markham[†], and Niki Trigoni[†]

[†]Department of Computer Science, University of Oxford, United Kingdom

[§]Department of Computer Science, University of Warwick, United Kingdom

[¶]Data61, CSIRO, Australia

ABSTRACT

This paper proposes *DeepAuth*, an in-situ authentication framework that leverages the unique motion patterns when users entering passwords as behavioural biometrics. It uses a deep recurrent neural network to capture the subtle motion signatures during password input, and employs a novel loss function to learn deep feature representations that are robust to noise, unseen passwords, and malicious imposters even with limited training data. DeepAuth is by design optimised for resource constrained platforms, and uses a novel split-RNN architecture to slim inference down to run in real-time on off-the-shelf smartwatches. Extensive experiments with real-world data show that DeepAuth outperforms the state-of-the-art significantly in both authentication performance and cost, offering real-time authentication on a variety of smartwatches.

Author Keywords

Authentication; security; smartwatch; split-RNN

ACM Classification Keywords

H.5.2. Information Interfaces and Presentation: User Interfaces - Input devices and strategies

INTRODUCTION

Smartwatches have been gaining great upsurge in public acceptance since their inception a few years ago: according to a recent report [1], 3.9 million Apple Watches have been shipped in Q3 2017, while the projected market volume will reach 86 million in 2021. Recent models including Apple Watch 3 and Samsung Gear S2 are equipped with a variety of sensors and built-in WiFi/cellular connectivity, and offer the same level of capabilities with smartphones. They can already make phone calls, reply emails, or pay without the presence of paired smartphones [2]. The new functionalities and independence have greatly improved their usability: now smartwatches are no longer complementary accessories tethered to phones, but independent components in the mobile and wearable world.

[*]Corresponding author: hongkai.wen@dcs.warwick.ac.uk

ISWC '18, October 8–12, 2018, Singapore, Singapore

© 2018 ACM. ISBN 978-1-4503-5967-2/18/10…$15.00

DOI: https://doi.org/10.1145/3267242.3267252

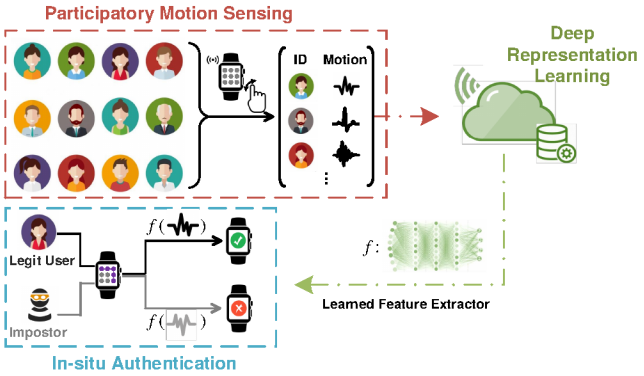

Figure 1: Workflow of DeepAuth

Unfortunately, unlike modern smartphones which typically employ strong biometrics such as fingerprints or face ID for user authentication, until now most commercial smartwatches only use PIN or Android Pattern Lock (APL) as passwords. A recent patent [3] considers vein structure as biometrics, but it relies on particular IR sensors, which may consume more energy and are not widely available in commercial devices. Voiceprint can provide convenient and fast user authentication, but it may not be user-friendly in many scenarios such as during meetings. On the other hand, behavioural biometrics, i.e. the ways users interact with devices, offers a promising option, but it is very challenging to make such authentication practically useful on smartwatches. One major reason is that it is often harder to learn behavioural biometrics such as motion signatures [5] on smartwatches due to hardware limitation, e.g. smaller screens and noisier sensors. Another fundamental challenge is that behavioural biometrics are typically learned from limited training data, and therefore tend to work well for previously trained users, but often fail when unseen imposters imitate the legitimate users.

To address these challenges, we propose a novel DeepAuth framework (as shown in Fig. 1), which exploits motion signatures when users entering passwords on smartwatches as behavioural biometrics, to provide a natural way of authentication in addition to the traditional APL. The proposed DeepAuth uses a deep neural network to model the input motion data, and considers a novel loss function to learn the optimal feature representations which are robust to noise, and can reliably reject unseen attackers with limited training data. Last but not least, DeepAuth offers in-situ real-time authentication

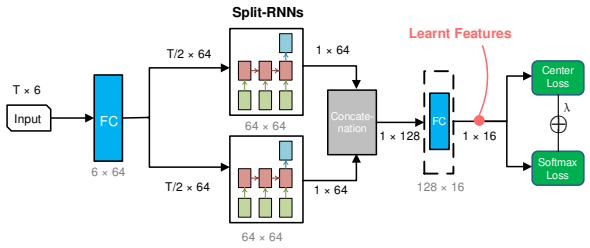

Figure 2: Proposed split-RNN layer (bottom). T is a variable denoting the length of an input motion sequence.

on off-the-shelf smartwatches, by slimming the proposed deep network and parallelising inference.

THE DEEPAUTH FRAMEWORK

The proposed DeepAuth framework consists of three major components, as shown in Fig. 1. In the *participatory motion sensing* module, smartwatch users voluntarily contribute their motion data to DeepAuth when entering passwords, in exchange for more secure authentication. Such crowd-sourced data, together with the associated user IDs are then used by the *deep representation learning* module, to learn an optimal feature extractor that can best distinguish the password input behaviour of legitimate users from attackers. Finally, the learned feature extractor is used by the *in-situ authentication* module, which runs as a daemon on smartwatches and authenticates the users when they enter passwords.

Participatory Motion Sensing

To make sure we obtain data from legitimate users, DeepAuth only initialises data collection when the smartwatches are in *authenticated states*, e.g. paired with the trusted smartphones. In this way, we implicitly leverage the strong authentication capabilities of smartphones, e.g. fingerprint or face recognition, to bootstrap DeepAuth. When detecting such an event, we sample both accelerometer and gyroscope to extract motion data segment pertaining to that password input. The data and associated user identity are uploaded to the cloud for further learning. DeepAuth only samples and transmits motion data, and need **not** to know the actual passwords. In OS level, e.g., Android, sending motion data from device to cloud without compromising the actual passwords is cheap and safe.

Deep Representation Learning with Limited Data

With motion sensing, DeepAuth obtains a set of motion data from different users when they input passwords on smartwatches. Each data point is a pair $(\mathbf{x}, y)$, where $\mathbf{x}$ is the motion data and y is its corresponding identity label. DeepAuth aims to learn a feature extractor f, which maps the motion data $\mathbf{x}$ to a lower-dimensional feature representation $\mathbf{f}_\mathbf{x} = f(\mathbf{x})$, and further to user identity y_i under a certain model.

Ideally, we would like f to be *password agnostic*, and *robust against unknown imposters*. The former requires that, given new motion data $\mathbf{x}^*$ entered by user y_i, which is generated from an *unseen* password, the extracted feature $\mathbf{f}_{\mathbf{x}^*}$ should still be mapped to identity y_i. More importantly, when a malicious attacker mimicked the legitimate user y_i to input the stolen password (generating motion data $\tilde{\mathbf{x}}$), we require that in the feature space, $\mathbf{f}_{\tilde{\mathbf{x}}}$ shouldn't be mapped to identity y_i.

However, learning such an extractor $\mathbf{f}$ is very challenging, especially given limited training data available: it is not possible to obtain motion data of all password combinations, nor any prior data from the unknown imposters. To address this, DeepAuth employs a deep Recurrent Neural Network (RNN) for feature learning, and considers a novel composite loss to enable the network to work with limited training data. In the following, we first briefly explain the RNN used in DeepAuth, and then show how we design appropriate loss functions to learn the optimal feature representation.

Deep Representation Learning: In DeepAuth, we consider a many-to-one network architecture, where the input motion data is firstly pre-processed by a fully-connected layer, and then fed into a RNN layer. This RNN layer can be implemented in different ways. Fig. 2 shows our split-RNN model (discussed later in Sec. 2.3). The output of the RNN layer is forwarded to a fully-connected bottleneck layer, and then to the output supervised via loss functions. We extract the activations of the bottleneck layer as the learned features $\mathbf{f}$, because by design it is compact in size, and should encode sufficient information since it is the last fully-connected layer before output. Now we explain how to train the above network to obtain the optimal feature extractor.

Composite Loss: To make the learned features password agnostic, DeepAuth prepares the training data by indexing over identity labels, i.e. for a given user y_i, all her motion data is fed into the training process directly, regardless of the password contents. This allows the network to pick up common patterns when a user enters different passwords, and produces password neutral features. More concretely, given the training data of m samples collected from g users ($y_i \in \{1, \ldots, g\}$, we use a combination of softmax loss and center loss to train a deep recurrent neural network:

$$\begin{aligned}\mathscr{L} &= \mathscr{L}_{softmax} + \lambda \mathscr{L}_{center} \\ &= -\sum_{i=1}^{m} log \frac{e^{W_{y_i}^T \mathbf{f}_i} + b_{y_i}}{\sum_{j=1}^{g} e^{W_j^T \mathbf{f}_i} + b_j} + \frac{\lambda}{2} \sum_{i=1}^{m} \|\mathbf{f}_i - \mathbf{c}_{y_i}\|_2^2 \end{aligned} \quad (1)$$

$\mathbf{f}_i \in \mathbb{R}^n$ is the extracted feature of the i-th input sample and current network. W and b are weights and bias are learn-able parameters the network. $\mathbf{c}_{y_i} \in \mathbb{R}^n$ be the centre of features for label y_i, which will updated as well during network training. The rationale of the centre loss term is that we found that $\mathbf{f}$ learned in practice can successfully distinguish between the known users, but is not robust to unknown imposters. As shown in Fig. 3(a) and (b), features of different users (points) are separable, but the clusters are not compact enough to reject potential imposters (see Fig. 3(b)). This is because by using softmax loss we implicitly train the network only for *classification* within labels $\{y_i\}$, but not extrapolation. To address this, we introduce a centre loss function in the training process, to pull the learned features towards their centres. Centre loss is proved to be effective in clustering face images [6] and enhance intra-class compactness. Therefore in DeepAuth, we propose to use the *composite loss function* in training deep

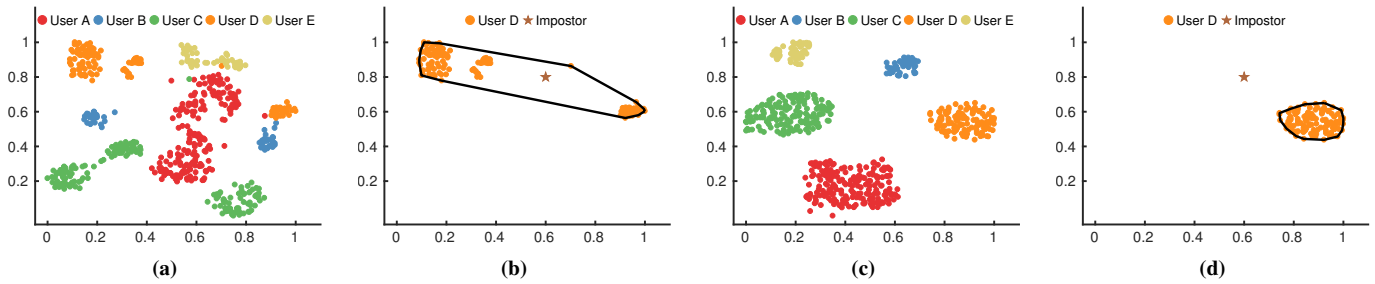

Figure 3: t-SNE visualisation of features learned using only softmax loss (a, b), and the proposed composite loss (c, d)

recurrent neural network. The hyper-parameter λ determines the trade-off between softmax and centre loss, and is obtained via cross validation. Fig.3 (c) and (d) show the effect of using composite authentication loss.

In-situ Authentication on Smartwatches

DeepAuth deploys the learned feature extractor f to the users' smartwatches, which is used to authenticate users from malicious imposters. For a user y (who may not appear in training), DeepAuth firstly builds a behavioural model locally, which encodes her unique motion signatures when entering passwords. Concretely, it collects password input motion data in authenticated states, e.g. when the smartwatch is paired with her own smartphones, and fits the extracted features with a multivariate normal (MVN) model: $\mathbf{f}_y \sim \mathcal{N}(\mu, \Sigma)$. When a correct password has been input on her smartwatch, DeepAuth extracts the feature $\mathbf{f}_{\mathbf{new}}$, and evaluates its distance from the user's behavioural model:

$$d(\mathbf{f}_{\mathbf{new}}, \mathbf{f}_y) = \sum_{j=1}^{n} \frac{\|\mathbf{f}_{\mathbf{new}}(j) - \mu(j)\|}{\Sigma(j,j)} \qquad (2)$$

Here we assume that individual feature elements are independent. If distance $d(\mathbf{f}_{\mathbf{new}}, \mathbf{f}_y)$ is below a certain threshold, DeepAuth accepts that the password was indeed entered by the legitimate user, and otherwise rejects this attempt.

Efficient Inference with split-RNNs: A key step of the above authentication process is to compute the feature $\mathbf{f}_{\mathbf{new}}$ from the observed motion data in real-time on the user's smartwatch. This is particularly challenging since resources (both computation and memory) on smartwatches are much more constrained than other platforms, and standard RNN inference is not feasible as it requires recursive processing of the input sequence.

However, we observe that in our context it is not always necessary to perform inference over the full sequence, because the correlation between the head and tail of the input may not be significant. For instance when entering an APL, the last few digits won't depend much on what was entered at the beginning. Therefore, it is possible to break the long input sequence and parallelise inference. Based on this intuition, DeepAuth splits the standard RNN model and distributes the inference task across two *split-RNNs*, as shown in Fig. 2.

The benefits are two-fold. Firstly, the RNN model size is reduced significantly, since the weight matrices in split-RNNs are halved, resulting in a much lower memory footprint. More importantly, inference on split-RNNs can be performed in parallel, and is more efficient since nearly half of the computation can be avoided. As shown later in Sec. 3.2, by using split-RNNs, DeepAuth can achieve real-time authentication on off-the-shelf smartwatches (<0.5s).

Methods	User Ratio	Precision	Recall	F1 Score	Accuracy
DeepAuth	0.2	0.75	0.77	0.76	0.92
	0.4	0.85	0.84	0.84	0.95
	0.6	0.87	0.85	0.86	0.96
	0.8	0.90	0.88	0.89	0.97
DeepAuth-Softmax Loss	0.2	0.67	0.84	0.71	0.87
	0.4	0.71	0.86	0.76	0.89
	0.6	0.78	0.87	0.81	0.93
	0.8	0.79	0.88	0.82	0.93
ICNP14	0.4	0.64	0.85	0.65	0.79
	0.8	0.72	0.91	0.77	0.89
MobiCom13	0.4	0.69	0.87	0.73	0.87
	0.8	0.75	0.87	0.80	0.92

Table 1: Authentication performance of DeepAuth and competing approaches.

EVALUATION

We evaluate the proposed DeepAuth framework extensively on large-scale real world datasets, collected in three different sites: *Oxford*, *Shanghai* and *Harbin*[1].

Experiment Setup

Online Password Survey: We consider Android Pattern Locks (APL), which are the default authentication approach on most of Android smartwatches. To make sure DeepAuth works with realistic passwords, we have surveyed 112 anonymous users online, and together with the results from [4], we built a set of 64 mostly used APLs, which are expected to be selected as passwords by ∼30% of users.

Data Collection: We recruited 155 participants (38% female) from the three sites, with age ranging from 20 to 35. Each participant is given 6 APLs randomly selected from the above 64, and is asked to enter them on a smartwatch worn on left wrist, for multiple times across different days. In total, we have collected 27,145 valid samples, each of which contains the motion data segment, and anonymous user identity.

Competing Approaches: We compare DeepAuth with three competing approaches: `DeepAuth-Softmax`, which is a naive version of DeepAuth using only softmax loss; `ICNP14` [7]; and

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

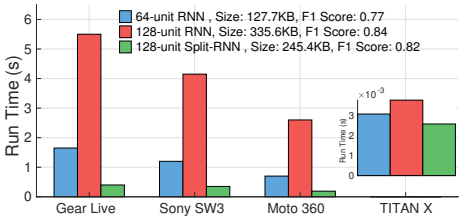

Figure 4: Efficiency and model sizes of different RNN layers across three smartwatch platforms and a desktop GPU. Model sizes and F_1 score are on the legend.

Mobicom13 [5]. The latter two are the state-of-the-art smartphone authentication approaches using behavioural biometrics. We port their implementations to smartwatch platforms and trained on our data. Unlike DeepAuth, they use handcrafted features with shallow classifiers such as SVMs.

Metrics and Evaluation Protocol: We evaluate the competing approaches with the following metrics commonly considered in existing work [7, 5]: *accuracy, precision, recall* and *F1 score*. For each subject, we randomly split all her instances of 6 APLs into a training and a test set, at the ratio of 7:3. Then in evaluation, we mix both seen and unseen passwords (outside 6 APLs of the subject) entered by imposters, and examine the authentication performance for every subject. We guarantee that each subject has data in both training and test sets, and 30% data in her test set are labelled as the positive and the data entered by impostors are marked as the negative.

Implementation: We train DeepAuth in an end-to-end manner with Adam optimizer. The initial learning rate is set to 10^{-3} and the hyper-parameter λ is set to 0.001. The batch size in training is 512. In order to avoid overfitting, we implement dropout in every fully connected layer and the dropout ratio is set to 0.2. All hyper-parameters, including the threshold of MVN model, are determined on a held-out validation set (15%) from the training set.

Results

Overall Authentication Performance: We first evaluate the overall authentication performance of DeepAuth and the competing approaches with respect to different amount of training data. We alternate the ratio between users and imposters in training and testing sets, from 0.2 meaning that only data from 20% of the participants is used for training, while the rest 80% is considered as unknown imposters in the test set, to 0.8 which is the other way around. In Tab. 1, we see that overall DeepAuth is able to achieve much better performance even with limited training data. On the other hand, using only the softmax loss (DeepAuth-Softmax) doesn't perform well: the F1 score when facing only 20% imposters is inferior to that of DeepAuth with 60% imposters. In addition, we see that approaches using handcrafted features (ICNP14 and MobiCom13) generally perform worse than DeepAuth. When facing 60% imposters, their F1 scores are far inferior to DeepAuth's. DeepAuth outperforms them in facing 60% imposters, even if competing approaches only faces 20% imposters. The significantly lower precision scores make competing approaches impractical for authentication. This means that they struggle in distinguishing legitimate users from imposters.

In-situ Authentication on Smartwatches: Now we evaluate the performance of DeepAuth when executed on off-the-shelf smartwatches. We deploy DeepAuth on three different watch models, including Sony SW3, Samsung Gear Live and Moto 360 Sports. We consider three variants of DeepAuth with different RNN implementations: 64-unit and 128-unit standard RNN, and the proposed 128-unit split-RNN (see Fig. 2). For all variants, the feature extractor is trained with the composite loss, on datasets with 0.4 user/imposter ratio. As shown in Fig. 4, we see that although on desktop GPUs (TITAN X) the inference time is comparable, on resource constrained smartwatches, standard RNN may take more than 5s to process one authentication request, which is clearly not practical. On the other hand, the proposed 128-unit split-RNNs can speed up inference up to ten-fold compared to the 128-unit standard RNN, and is even 3-4 times faster than the 64-unit RNN, due to the effective parallelisation and reduced computation. This is because for both 64-unit and 128-unit RNNs, the computation is sequential while with 128-unit split-RNN, we can parallel the computation of two sub sequences halved from the original one. In addition, as shown in lengend of Fig. 4 the size of split-RNN is $\sim$30% smaller than the standard 128-unit model, while the F1 score is comparable (only 2% lower). Above results imply that DeepAuth with split-RNNs can reduce both inference time and memory footprint significantly, offering real-time authentication on smartwatches.

CONCLUSION

In this paper, we presented DeepAuth, a novel authentication framework for smartwatches based on behavioural biometrics. DeepAuth uses a novel slimmed deep neural network with composite loss functions, to learn robust features from noisy motion data across different users, which can run in real-time on resource constrained smartwatch platforms. Extensive experiments with real-world data confirm that DeepAuth is able to provide a natural and user-friendly authentication mechanism on smartwatches in addition to traditional passwords, and can achieve impressive performance against unseen attackers even with limited training data.

REFERENCES

1. 2017. Media alert: Apple retakes the lead in the wearable band market in Q3 2017. (2017).
2. Ltd Alipay.com Co. 2017. Alipay - Makes Life Easy. https://itunes.apple.com/us/app/alipay-makes-life-easy/id333206289?mt=8. (2017).
3. H.S. Kim and et al. 2017. Wearable device and method of operating the same. (2017).
4. Marte Dybevik Løge. 2015. *Tell Me Who You Are and I Will Tell You Your Unlock Pattern*. Master's thesis.
5. Muhammad Shahzad and et al. 2013. Secure unlocking of mobile touch screen devices by simple gestures: you can see it but you can not do it. In *MobiCom*.
6. Yandong Wen and at al. 2016. A discriminative feature learning approach for deep face recognition. In *ECCV*.
7. Nan Zheng and et al. 2014. You are how you touch: User verification on smartphones via tapping behaviors. In *ICNP*.