Securing smartphone handwritten PIN codes with recurrent neural networks

Gaël Le Lan Vincent Frey

Orange Labs, France

orange™

Abstract

- PIN code based authentication on smartphones
- Digits handwritten by the user (no keypad)
- User authenticated through his/her writer traits
- RNN as a discriminative feature extractor
- Evaluations run on two datasets of 43/33 users
- 4.9% EER for a 4-digits PIN code
- Digit value prediction during training is key

Objectives

- Enforce the knowledge factor (e.g. password or PIN code) with behavioral biometrics
- Authenticate users through their writer traits

Authentication System

- Smartphone application (OTP scenario)
- Enrollment: 4 examples of each handwritten digit in a local template database
- Trial compared with examples stored in template (1-nearest neighbor)
- Decision based on writer traits recognition (threshold on trial/template similarity)
- Attack scenario: all impostors know the PIN code digits

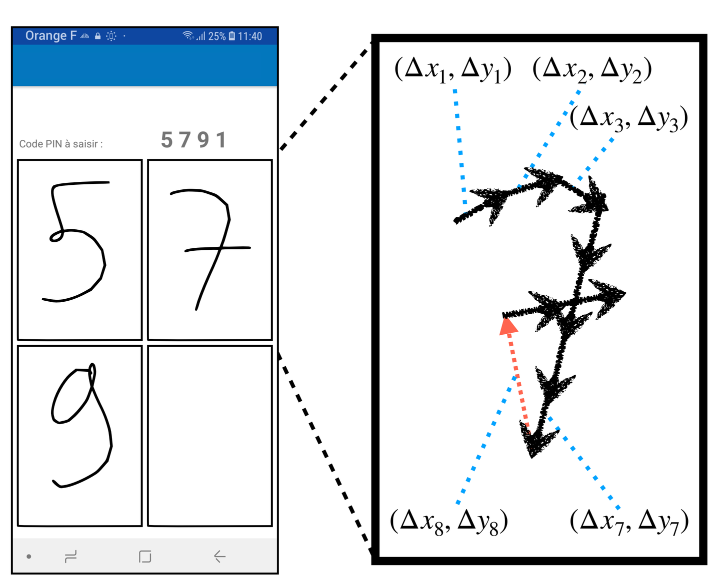

Figure 1: Application interface

How to model writer traits ?

- Each drawn digit: variable length sequence $\boldsymbol{S}$ of strokes $\boldsymbol{s}_i = (\Delta x_i, \Delta y_i)_{i \in [1..N-1]}$
- Encode $\boldsymbol{S}$ into a representation $f(\boldsymbol{S})$ of fixed dimension
- f: bidirectional Recurrent Neural Network
- Compare digits (sequences) $\boldsymbol{S}$ and $\boldsymbol{Q}$ through cosine similarity

$$sim(\boldsymbol{S}, \boldsymbol{Q}) = \frac{f(\boldsymbol{S})f(\boldsymbol{Q})}{\|f(\boldsymbol{S})\|\|f(\boldsymbol{S})\|} \quad (1)$$

- RNN trained to predict writer identity and digit value on a *train* dataset using cross entropy loss

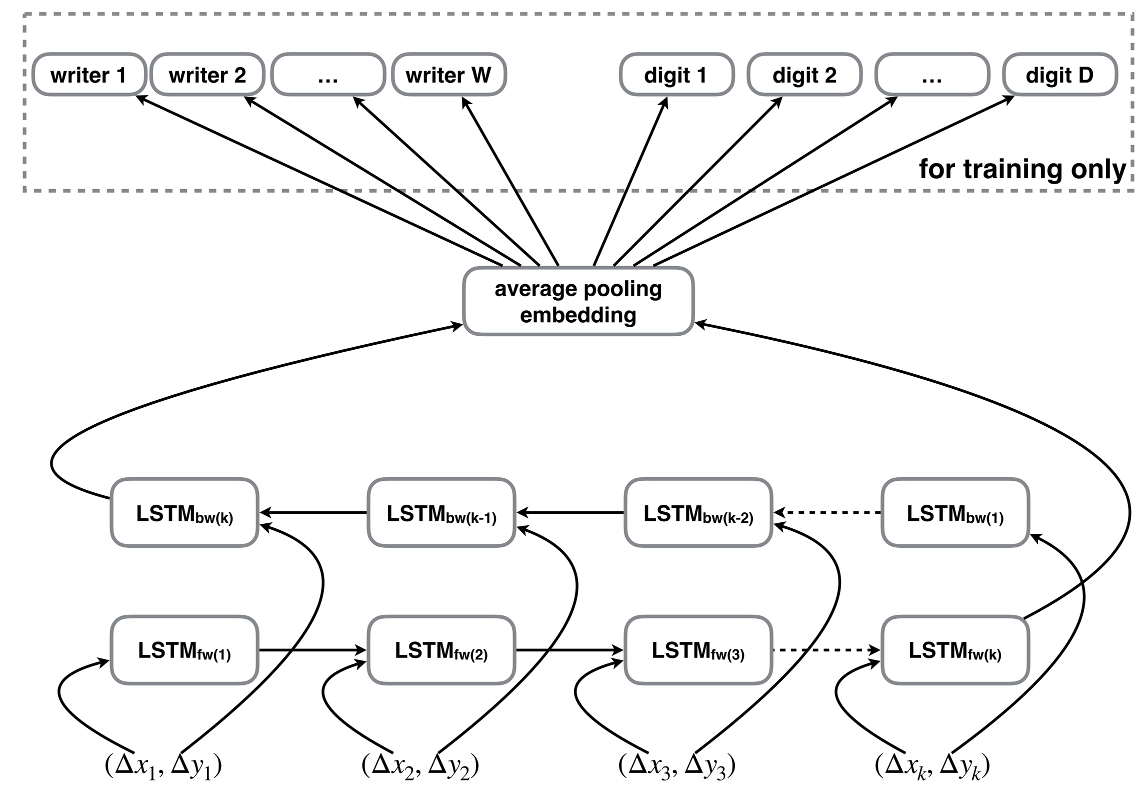

Figure 2: Overview of RNN architecture

$$\mathcal{L}(\mathcal{T}) = -\frac{1}{M}\sum_{k}\left[\sum_{w=1}^{W} \mathbb{1}_{[\boldsymbol{S}_k \in w]} \log(p_{\boldsymbol{S}_k \in w}) + \sum_{d=1}^{D} \mathbb{1}_{[\boldsymbol{S}_k \in d]} \log(p_{\boldsymbol{S}_k \in d})\right] \quad (2)$$

Key concepts

- Bidirectional RNN trained to encode discriminative digit representations
- After training, used as a feature extractor for any user
- Embedded in the smartphone using Tensorflow for Android
- Cosine similarity for trial/template comparison between digit representations
- Evaluation on users unseen during RNN training
- Effective writer traits modeling on unseen symbols

Table 1: System performances on various *train/eval* combinations.

#	system	train	train sessions count	eval	EER with digit prediction 1 digit	EER with digit prediction 4 digits	EER without 1 digit	EER without 4 digits
1	[1]	$eBioDigit_{train}$	4000	$eBioDigit_{eval}$	-	-	18.6	9.3
2	proposed				15.1	6.5	20.9	11.3
3				$internal_{eval}$	18.5	8.2	23.1	13.2
4		$eBioDigit_{train} + internal_{train+eval}$	8 750	$eBioDigit_{eval}$	**12.5**	**4.9**	**18.0**	**9.9**
5		$eBioDigit_{train+eval} + internal_{train}$	8 890	$internal_{eval}$	15.8	6.3	22.7	11.8

Datasets

Taken from *eBioDigit* and *internal* evaluation campaigns.

Table 2: Datasets composition.

name	*eBioDigit* [1]		*internal*	
device	Samsung Galaxy Note 10.1		Samsung Galaxy A8	
type	*train*	*eval*	*train*	*eval*
users	50	43	29	33
sessions/user	2		1	2
digit/session	10×4		10×5	

Experimental setup

- For each *eval* user, 2 sessions of data collection
  - First session for enrollment
  - Second session to simulate trials
- Equal Error Rate computation over all possible genuine/impostor trial/template scores

Future work

- Expand to other symbols: letters, drawings
- Other source of information: accelerometer, gyroscope, finger pressure...
- *Shoulder surfing* attack scenario: impostors can see how genuine users draw their digits

[1] Ruben Tolosana, Ruben Vera-Rodriguez, Julian Fierrez, and Javier Ortega-Garcia, "Incorporating touch biometrics to mobile one-time passwords: Exploration of digits," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 471–478.