Biometric Unlocking System

Jianxiong Gao
University of Illinois
102 Sterling CT
Savoy, IL 61874
gao2@illinois.edu

Junle Qian
University of Illinois
606 Stoughton ST
Champaign, IL 61820
qian10@illinois.edu

ABSTRACT
Mobile devices has becoming more and more ubiquitous in our modern lives. People tend to have frequent interactions with their mobile devices, such as cellphones and tablets. While mobile devices interact with human, there is a great potential for them to learn owner-specific information. Thus device security has become a more serious issue than before. In this work we present BUS, a biometric information sensing system that identifies the legitimate user from behavioral information. When the user uses a smart phone, our phone application works in the background and learn the behavior of the user. After some training data has been obtained, BUS starts to detect whether the current user is legitimate and decide if particular permissions should be granted, such as unlocking the screen of the device. The information we collect is solely based on touchscreen input, in the consideration of power limitation. All the training and detection phase are done on the smart phone in an offline mode, thus no privacy issue needs to be considered.

Categories and Subject Descriptors
K.6 [**Security and Protection**]:

General Terms
Identification, Touch, Behavioral Biometrics, Security

1. INTRODUCTION
With the rapid development in mobile device industry, personalized apps on mobile devices has dramatically changed our lives. However, while these apps get more involved in our lives, there is more personal information stored on the device. The owner of the device faces an increasing risk of privacy leakage not only from the system or apps, but also from sharing device with guest users, such as family members, friends, partners or coworkers [2]. Thus user identification is necessary to protect the privacy of the smart phone user. User identification has been a popular topic in security area for a long time. There are three general categories of information that is used to verify an identity in computer security: [7]

- What you know(password)
- What you are(fingerprint)
- What you have(driver's license)

The traditional password is simple to set and use, but it is also likely to be compromised. Anyone who watches the user typing the password can hack in successfully. The fingerprint scanner used on the latest *IPhone 5s* smart phone is securer than a password or pin. However, the fingerprint information can not be reset once it is compromised. There are various concerns on using iris or fingerprint as password because of this reason. Using the *what you have* information for user identification adds too much complication to the process, thus it might not be applicable in daily use of the smart phone.

Researchers have proposed several different types of user authentication system utilizing an abundant amount of sensors on the smart phone. However most of the proposed work utilizes several different sensors on the mobile device to improve the accuracy. [5] [2] Utilizing more sensors not only consume more power, but also generates more data which requires more computation time and storage space. Our BUS system is a light weight application which requires only touch screen input information. Thus we can limit the power consumption and computation overhead of user identification to a minimum, while achieving reasonable accuracy. In reality we can recognize the legitimate user for more than 80% of the chance, while potential attackers who observe the entire unlocking process has to try for more than 10 times to successfully break into the system.

This paper is based on the following structure, section 2 is a basic description of support vector machine which we use as our machine learning engine, and how we select the *feature points* for our system. Section 3 describes the process of learning and detecting the unauthorized behavior. We summarize our results and conclude in section 4.

2. SUPPORT VECTOR MACHINE
Before mobile devices has become such a popular tool in our daily life, the problem of identification has been existing for nearly the whole human history. The method we used to

solve the problem is by using signature. The nature of the question of signature verification is similar to the identity verification of smart phone user. In recent 20 years, there are numerous machine learning algorithms proposed for automated signature verification.[6] [4] There are three major kinds of machine learning engine that are mostly used in these literatures:

- Hidden Markov Model
- Support Vector Machine
- Neural Network

There are other techniques used in addition to these engines, such as dynamic time warping. However on a mobile platform we need to consider power and computation time related issues. Thus after some probing tests we selects support vector machine as our engine because it is the most light weighted engine. The learning process required for the support vector machine is shorter compared to the other two engines. However, utilizing support vector machine, we need to carefully extract our *feature points* that we feed into the model, in order to get the best result. In the latter part of this section we first introduce support vector machine and then describe how we select the desired *feature points*.

2.1 Support Vector Machine

Support vector machine has been a popular classifier in machine learning area for more than two decades. The general approach for utilizing support vector machine has the following procedure:

1. Cross validation to find the best parameter
2. Use the best parameter to train the support vector machine with the whole dataset
3. Apply the test dataset to the trained model to determine the classification

There are several models that can be used for support vector machine. The library we choose to use is the most popular support vector machine library: [3] . It provides 5 different models including:

- linear
- polynomial
- radial basis function
- sigmoid

After some experiment, the model we choose to use is the radial basis function model. This model consider all the features we feed to the support vector machine equally, which is what we desired. There may exist other better models but that is left as future work.

With limited number of training sets, there may be numerous ways to set the classifier. The actual resulting classifier by radial basis function is determined by c and γ in our case. By figure 1 we can see two different valid classifiers under the same input *feature points*. To find the best c and γ value, we use the cross validation method to compute a predict accuracy. Cross validation is a process of grouping several sets *feature points* as test sets, then use the rest of the sets as training data. After training, we feed the test sets back to test if the prediction is correct. Finally we produce an prediction accuracy for the c, γ value provided.

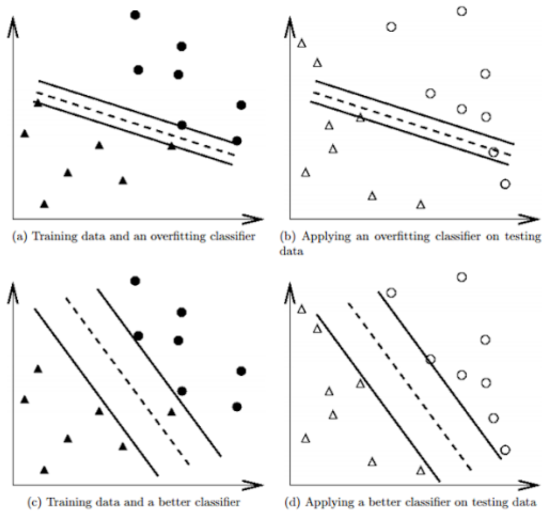
(a) Training data and an overfitting classifier
(b) Applying an overfitting classifier on testing data
(c) Training data and a better classifier
(d) Applying a better classifier on testing data

Figure 1: Difference between classifiers

The number of set of *feature points* has great influence on the result accuracy. Several literatures indicate that around 16 sets is the optimal number.[1] Our test result verified this. With not enough training the resulting classifier can be very random. With too much training data on the other hand, gives worse overall result. One explanation we have is that the biometric feature of human changes over time. Too much history may produce a sparser matrix of *feature points*, which lead to a worse result.

2.2 Feature Points

While support vector machine adds little power and computation overhead to the system, it is not as powerful as other machine learning engines. The most important part of utilizing support vector machine is the what feature points we feed into it. In our design, in addition to the static graphical information, we would like to have the time information as input. In addition to this, because we need to keep around 16 feature points for each behavioral pattern for future cross reference, we have to keep each set of *feature points* small such that we can keep all the sets in a relatively small size. After experimenting with several different approaches, our finalized version traces all the strokes, from the initial touch until the last release of the finger. Then the strokes are then divided into 30 segments according to timing information, and the location information at each segment point is recorded. In addition to the location information, the curvature information is also recorded. In total these come up to a sum of 120 integer information. In addition to this, we further define that each touch until release is a path, and record this information with the total time information. Thus for each *feature point* we have a total of 122 integer information as listed below:

- 30 x location information
- 30 y location information
- 30 x curvature information
- 30 y curvature information
- 1 number of path information
- 1 total time information

We further take location and size of the strokes. We take the lowest x,y value and subtract them from every (x,y) pair in the location information, and then normalize all the locations into a range (0,1). In this way the size and location of the strokes does not influence the effect of BUS.

3. LEARNING AND DETECTION

Ideally our BUS application would train itself through real user's daily usage. For example learn how the user type specific words when messaging others. However due to lack of time and permission from the android system, we can only build a prototype such that we can train it with our own predefined character. Because our BUS system utilizes a cross validation step while training, and the cross validation step results in a confidence level for the trained model. We argue that with enough user input we can generate a reasonable amount of sets of *feature points*, from which we select several with higher confidence level as our detector model. In the prototype now this step has been omitted, we manually selected several character as input to the system.

We first train the system with the input of our own, with a label of which character it is. The training takes around 16 times as indicated previously. The training interface is shown in figure 2.

After training with several symbols, we can go into the unlock interface. The symbol needed to unlock the system is chosen randomly from all the trained symbols. The successful try and unsuccessful try example can be found in figure 3 and figure 3 respectively.

4. CONCLUSIONS

With real user testings, the owner can easily achieve a successful unlocking rate above 80% of the time, while the other tester watches the whole process and try the same procedure, the successful rate is well below 10% without any training. Even with some training the successful rate of mimicking other users behavior is around 50%. All these results are based on a single symbol. We argue that if there are multiple symbols and we let the user to try to match randomly, the rate of successfully bypassing the system is even lower. Furthermore, if we can monitor the real user behavior, it will be nearly impossible for the attacker to mimic all the behavior of the real owner.

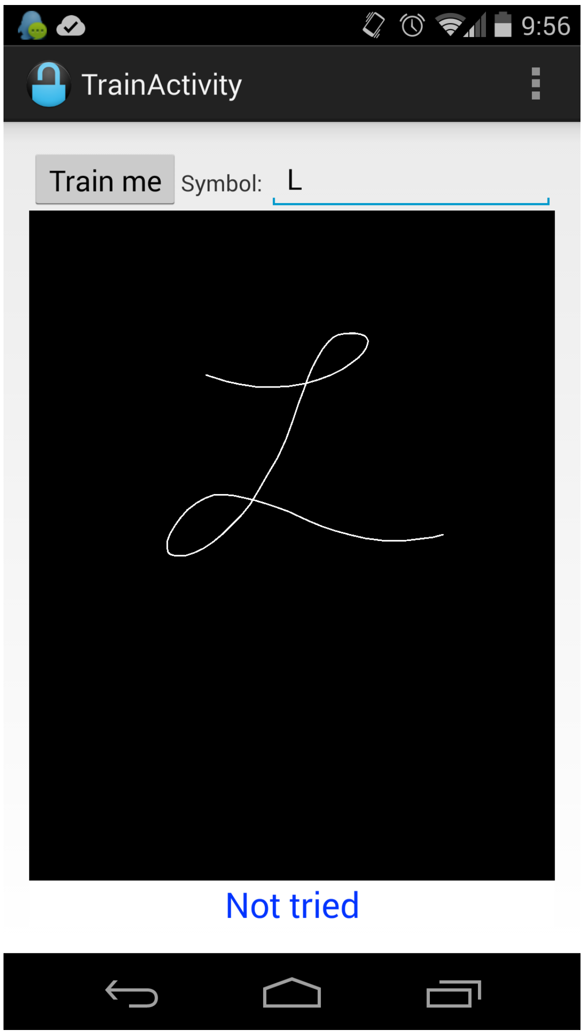

Figure 2: Training of the letter L

5. REFERENCES

[1] S. Audet, P. Bansal, and S. Baskaran. Off-line signature verification using virtual support vector machines. *Artificial Intelligence, Final Project*, 2006.

[2] C. Bo, L. Zhang, and X.-Y. Li. Silentsense: Silent user identification via dynamics of touch and movement behavioral biometrics. *arXiv preprint arXiv:1309.0073*, 2013.

[3] C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[4] A. Julita, S. Fauziyah, O. Azlina, B. Mardiana, H. Hazura, and A. Zahariah. Online signature verification system. In *Signal Processing & Its Applications, 2009. CSPA 2009. 5th International Colloquium on*, pages 8–12. IEEE, 2009.

[5] J. R. Kwapisz, G. M. Weiss, and S. A. Moore. Cell phone-based biometric identification. In *Biometrics: Theory Applications and Systems (BTAS), 2010 Fourth IEEE International Conference on*, pages 1–7. IEEE, 2010.

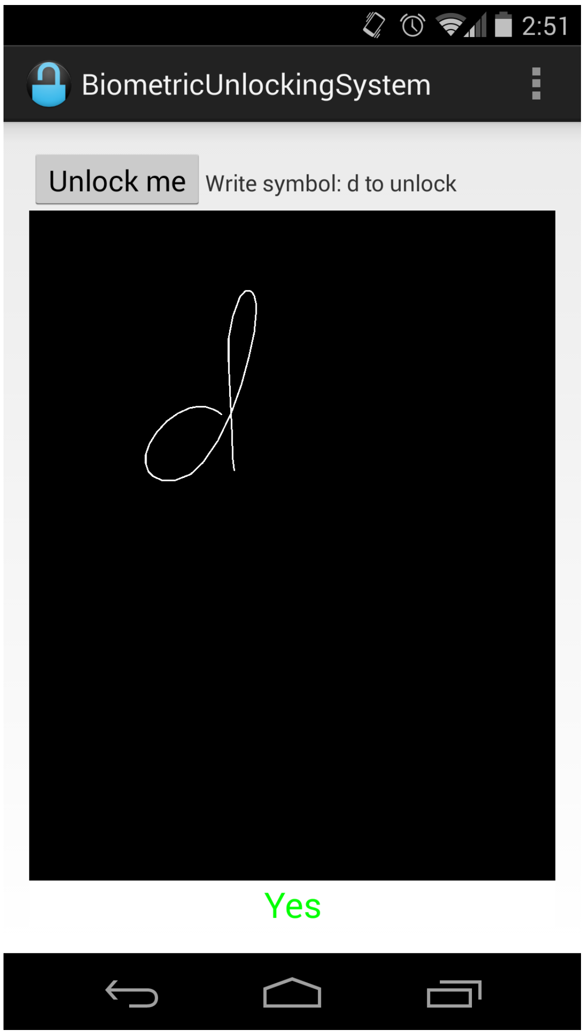

Figure 3: Successful Unlock

[6] S. Pal, M. Blumenstein, and U. Pal. Off-line signature verification systems: a survey. In *Proceedings of the International Conference & Workshop on Emerging Trends in Technology*, pages 652–657. ACM, 2011.

[7] W. Stallings and L. V. Brown. *Computer security*. Prentice-Hall, 2008.

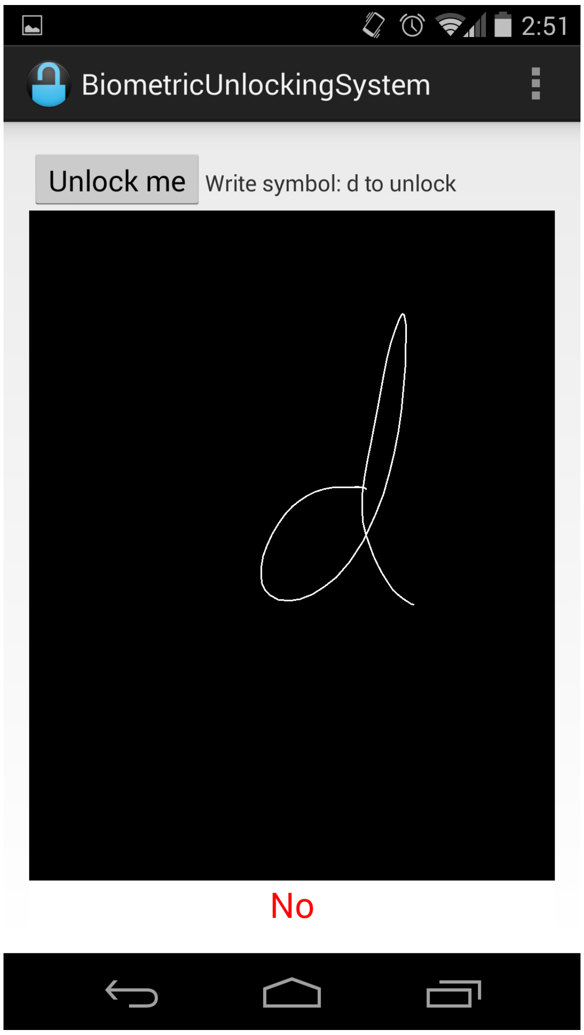

Figure 4: Unsuccessful Unlock