Keystroke Recognition Using Android Devices

João Lopes, Paulo Correia

Abstract - The term “biometrics” is derived from the Greek words “bio” (life) and “metrics” (to measure). Biometric recognition is therefore related with the recognition of people based on their characteristics. Automatic biometric recognition systems have become available over the last decades, due to significant advances in computation. However, until recently, specific devices were needed for biometric recognition. Nowadays, smartphones have a considerable processing power, allowing to implement some biometric algorithms. A demand for biometric recognition was created due to increased smartphone market penetration, since the devices hold sensible personal information. To have a secure access to sensitive information some type of security against illegitimate users is needed. Biometric security is therefore a must on these devices, given that the traditional PINs (Personal Identification Number) can be stolen, forgotten or cracked. On the other hand, personal characteristics are unique and can’t be forgotten and are hardly stolen, making biometric validation superior to PIN usage, creating a demand for biometric validation applications to secure people’s information. To increase security a PIN is commonly used together with biometric identification. The system proposed in this dissertation aims to monitor mobile phone users for a pattern while writing on keyboards (keystroke) and then using this pattern to secure the mobile phone from unauthorized users. The proposed system can use an algorithm based either on Euclidean distances or Support Vector Machines, for the classification stage. Encouraging results were obtained using the SVM classifier.

Index Terms—Biometric recognition, personal identification, keystroke dynamics, smartphone, Euclidean distances, Support Vector Machine

I. INTRODUCTION

Mobile phones have a central role in everyday life. Worldwide, the number of active cellphones now exceeds the world population, and the same penetration growth trend is observed in Portugal. Smartphones are in fact small computers and their operation is governed by an operating system. Since smartphones appeared and took charge of our information and communications, a need to enhance the security of these devices exists. Most people use a pattern (combination of movements that lock the phone screen) or a PIN to access the device, however they are easy to detect crack. However, adding a biometric trait to PIN’s provides an extra level of security.

This paper discusses the development and testing of an application for android OS smartphones, which performs biometric verification of the user based on the keystroke dynamics when entering a password. The developed biometric recognition system follows the general architecture represented in Figure 1. Based on that figure, the author has designed, implemented and tested a biometric verification system capable of identifying users. However, as the smartphones standard keyboard does not have some of the necessary functionalities for this work, another keyboard had to be developed in order to proceed with the remaining work. The corresponding software implementation has been fully developed by the author.

Figure 1- Generic architecture of a biometric recognition system

II. BIOMETRIC SYSTEMS

Biometric systems are automated methods that verify or recognize the ID of a person based on a physical, physiological or behavioral characteristics. When conjugated with traditional security methods they provide an extra level of security. Examples of biometric characteristics would be fingerprints, face, and iris, among others.

Enrollment is the process of collecting biometric data from a user and store it in the system. Furthermore, authentication is the identification or verification of the user’s identity by matching the data provided by the user with the data stored in the system. During the enrollment, the biometric system stores biometric traits of the user. During authentication, this traits are used to recognize a user who provides his biometric trait. Enrollment is an important step regarding the accuracy of the template, so it should be not limited to one-time step and keep updating the user template. As observed in Figure 1, a generic biometric system is assembled by 5 major components, input or biometric sensor, feature extraction, classification/matcher, decision maker and a template associated with the system. The first component, biometric sensor, is responsible for the scanning the biometric trait of the user, being the interface between the user and the authentication system. Next, feature extraction is responsible to extract salient data that is responsible to distinguish between different users. During the enrollment the data extracted is stored in a template. The matcher is a module that compares the input with the template and then indicates the similarity between those two. The decision module, makes the identity decision.

Any biometric system will exhibit occasional false acceptance of intruders and false rejection of legitimate users. The corresponding False Accept Rate (FAR) and False Reject Rate (FRR), as well as the Equal Error Rate (EER or CER, which stands to Cross-Over Rate), where FAR equals FRR, are important metrics to ensure the validation. FAR ought to be low, as it specifies the probability that an imposter can use the
device, as well as FRR, which can cause inconvenient when the ratio is high.

Besides the FRR and FAR metrics, there are others such as sensitivity, specificity and accuracy. These 5 metrics will be used to assess the obtained results, in section V. Sensitivity, also called true positive rate, measures the actual positives which are correctly identified, and it is complementary to the false negative rate. Specificity, also called true negative rate, measures the negatives which are correctly identified, being complementary of the false positive rate. Sensitivity and specificity can be calculated according to formulas (1) and (2), respectively. Accuracy, which can be calculated using equation (3), assesses how well the system behaves, and allows choosing the optimal operation threshold for a system.

\[
\text{Sensitivity} = \frac{TP}{TP + FN} \quad (1)
\]

\[
\text{Specificity} = \frac{TN}{TN + FP} \quad (2)
\]

\[
\text{Accuracy} = \frac{TP + TN}{\text{Total Population}} \quad (3)
\]

III. KEYSTROKE DYNAMICS

Keystroke dynamics recognition consists in the recognition of an individual based on the way he types, using a mobile keyboard. In particular when employing keystroke dynamics as a biometric trait, there are two major authentication strategies that can be employed: static or continuous. In static biometric authentication, each participant provides his biometric features during enrollment, which is stored in a template. Whenever a person tries to authenticate herself, she will provide a new sample of the same biometric feature and this new input is compared to the ones previously stored in the template. Authentication using static keystroke dynamics is based on measuring the duration of key presses by the user, and on the time latency between consecutive keystrokes, relating them as they are being pressed [1]. In continuous biometric authentication instead of typing a fixed text, the system is used with unconstrained textual input (free-text), typically for a longer period of time. Over that time, information is collected on how the user types on the keyboard, during enrollment. During authentication the features computed from the input text values are compared to the ones stored in the template [2].

Figure 2 represents the duration of a keystroke which is the time of the time when the key is pressed and when it is released.

A. Input sensor

A system relying on the analysis of keystroke dynamics needs to capture relevant information by accessing the user characteristics. As smartphones have software keyboards, only soft keys will be considered. A soft keyboard is a system that replaces the physical keyboard on a computing device with an on-screen image map. The touch screens, in these devices, used for displaying the keyboard and receiving the corresponding information, can be either resistive or capacitive. On resistive screens the pressure applied on the screen can be read, as they function based on finger pressure. Furthermore, capacitive screens can read the size of the surface of the finger, once they detect anything that is conductive. As summarized in Figure 3, the input can be done via a soft or hard keyboard.

B. Feature extraction

A feature is a distinctive attribute or characteristic of something. Every human has different characteristics or attributes and hence it is possible to distinguish, i.e., identify them. With that in mind, in this work, features will be defined such as the typing rhythm of the user.

There are different methods and metrics upon which keystroke analysis can be based [4]:

- **Static at login**: a known keyword, phrase or predetermined text is captured and then compared against stored typing patterns.
- **Periodic-dynamic**: the user typing pattern is captured during a part of a logged session and then compared against stored typing patterns to determine deviations.
- **Continuous-dynamic**: similar to the periodic dynamic but the authentication is done during the entire logged session.
- **Keyword-specific**: is an extension of continuous or periodic dynamic but related to specific keywords.
- **Application-specific**: continuous or periodic dynamic applied to a specific application.
- **Keyword latency**: consider the overall latency for a complete word.
Some additional features can be considered when using smartphones and tablets:

- **Pressure during typing**
- **Fingertip size**
- **Physics of the mobile device**: it means how the user holds his device and which is the preferred hand.

C. **Classification techniques**

Classification techniques enable a classifier to identify to which of a set of categories a new observation belongs. This is possible through the training of the algorithm on the basis of a training set of data containing observations whose category membership is known. A category would be composed by a set of features which defines it. An example would be assigning a password input into true or false user classes. In this context, there are two main classification approaches followed for keystroke analysis: statistical techniques and neural networks techniques or a combination of both [5]. Furthermore, both need a matcher and stored data, to allow the processing of the keystroke timings.

The essential features to be used for the classification step are keystroke timings, which are the timings between successive keystrokes, press and release events. The time between both events is called dwell time, on the other hand, the time between the release event and the press event of the next key is named flight time. The template to be used for recognition is constructed with basis on this concepts. The template refers to the process of determining, from a give set of available biometric acquisitions, which are the best suited to represent the collected data and the statistics of the considered users’ biometrics [6].

D. **Keystroke models**

Keystroke models authentication can be classified as either static or continuous. As it was referred previously, static authentication refers to keystroke analysis performed only at specific times, as during a login process. An example is PIN model [5], where the PIN number is introduced by the user several times during enrollment. The user timing vector is captured and enrolled in keystroke acquisition. Other keystroke features are extracted and their mean, standard deviation and median is calculated which is given as input to the feature subset selection.

In addition, continuous authentication performs the same analysis but during the whole session. This method provides a tool to also detect user substitution after successful login. The free-text model is a continuous authentication system, looking for the continuously presence of the authorized user. This is done by analyzing the typing rhythms user show during their normal interaction with a computer. There is a long time of data collecting due to many combinations of words. In [7] word specific digraphs are constructed from the most common words used, due to sample dispersion. They used monographs and digraphs analysis and a neural network to predict missing digraphs.

IV. **EXPERIMENTAL SYSTEM**

The main objective of this work was to develop an application for android OS smartphones, which performs biometric verification of the user based on the keystroke dynamics when entering a password. It is developed in the Java programming language using the Android Software Development Kit (SDK), which includes a comprehensive set of development tools including a debugger, a set of libraries, an emulator, documentation, sample code, and tutorials.

A. **Architecture**

Figure 4 illustrates the architecture of the projected system.

![System architecture](image)

In the first step, the input is an alphanumeric password, as the writing done in smartphones is relatively short making free-text input not the most suitable for smartphones. As the app deals with keystroke dynamics as a biometric trait, the classification algorithms used for the classification step need training. Due to that, the developed application has two operation modes. One is the training mode, in which the user inserts the chosen password to allow the system to capture the corresponding features and store them in a database. For the purpose of this work, to be able to compare different types of password, three passwords have been chosen, notably: mxplayer, Lisboa2014 and tecnicoLisboa. The second mode, verification, is the mode where the actual input (of the true user or the intruder) is compared against the data gathered during training. The app will allow different users to register and choose one password.

B. **Capturing user input**

At the same time each key is pressed or released the time metrics are measured and stored in a SQLite database, which is available from an android library. Each training operation has a different id, and each id has associated with a set of the key codes, dwell times and flights time for the chosen password. As the application records key timings, while the user is typing the password, if the user makes a typing mistake, that input will not be valid, because it would invalidate the training data as the key timing would be greater than expected.

To be able to capture key times, the developed application implements two functions `OnKeyDown` and `OnKeyUp` which handle key down and key up events, respectively, when they occur. Also a new soft keyboard was developed and included in
the application, to be able to control the generated press and release events.

C. Classification and decision

When the user enters the verification mode, he can choose which algorithm to use, Euclidean distances and SVM. On both of them, the user has to enter the same password one more time, to be able to verify if the user being tested is the true user.

When using Euclidean distances, a distance is calculated between the entered password and the ones stored in the database. That distance is calculated according to the formula in (4). The result from is stored in a vector that holds all the distances. Then, upon a threshold, the algorithm will decide if the user is valid or not.

\[
d = \sqrt{\sum (x_i - y_i)^2}
\]  

(4)

Regarding SVM, the implemented algorithm uses ‘libsvm’ for training and testing of the features. Before starting training the data, it has to be normalized. This is a common procedure in machine learning. The normalization consists in converting a vector into a unit vector, between 0 and 1. This trains the SVM on relative values of the features, not magnitudes. The procedure is done by dividing each value by the norm of the vector. The norm is calculated using the formula in (5).

\[
\sqrt{\sum x_i^2}
\]  

(5)

After this, the values are mapped into two hashmap’s, one for training labels and the second for training/test features. Then, a training vector for each password is created, which is composed by a number of nodes that belongs to one password. For the nodes training, there are some parameters that can be set, kernel type, parameter C and gamma. The kernel can be linear or nonlinear. The decision whether to use one or the other, has some facts take into consideration. Typically, the best possible predictive performance is better for a nonlinear kernel, or at least as good as the linear one. It’s been shown that linear kernel is a degenerate version from RBF (Gaussian), which is a nonlinear kernel, hence the linear kernel is never more accurate than a properly tuned RBF kernel. This affirmation is only not true when the number of features is large relatively to the number of samples, [8]. In that case is good enough to use linear kernel, because nonlinear kernels do not score better than the linear one. In the case of this work, the number of features is small as well to the number of samples. So, to sum up, the RBF kernel is the chosen one. With this kernel, there are two parameters that can be selected which is C and gamma. Parameter C tells the SVM optimization how much you want to avoid misclassifying each training set. For large values of C, the optimization will choose a smaller margin hyperplane, if it does a better job of getting all the training points classified correctly. The opposite happens to small values of C, the optimization will look for a larger margin hyperplane, even if that hyperplane misclassifies more points. On the other hand, parameter gamma should be chosen according to the magnitudes on the pairwise distances of the data points. If the value gamma is very small, RBF kernel is very wide, meaning all the data points could fall into one class. However, if gamma is very large, RBF kernel is very narrow, meaning that, probably, all training vectors will end up as support vectors. These two extreme situations are not desirable, so a combination between the two extremes should be encountered. An illustration of a SVM linear kernel can be seen in Figure 5.

\[
\text{Figure 5 - SVM linear kernel illustration [9]}
\]

The final results are obtained with a function, ‘libsvm’, which predicts a probability for each of the classes. Finally, upon a threshold, the user is validated or rejected.

V. RESULTS

In this section some results, from the two implemented algorithms, will be presented, as well as some discussion about the final results. One drawback from the analysis, is that the database does not have many users, making generalization rather limited. However there are 3 different passwords. These passwords were chosen carefully, so that each one has different characteristics: one is lowercase only, another includes lowercase, uppercase and numbers, while the third one includes lower and uppercase.

To analyze the performance of the algorithm and calculate the best operation thresholds for each user, a ROC curve should be plotted. A ROC curve is a graphical plot that illustrates a performance of a classifier as its threshold varies. The curve is plotted by the ratio between true positives and false negatives at various threshold settings. All ROC curves were plotted using XLSTAT software, which is an add-on for Microsoft Excel. This allows excel to plot ROC curves by giving the correct and incorrect values from the users. Subsequently, the threshold is chosen based on best accuracy for that ROC curve. However, depending on the goal of the verification, the threshold should be chosen accordingly. If the goal is to secure sensitive information, the threshold should be lower, to not allow false negatives. On the other hand, the threshold might be higher, so the user do not have to enter the password more than once.

A. Euclidean distances.

With this algorithm the threshold represents a distance, where the user is accepted when the distance of the password inserted is lower than the threshold calculated. In Table 1, the figures for the best accuracy achieved are represented. Despite the thresholds indicated in the table being an optimal solution,
there are other possible thresholds than can be used depending on the goal of the application.

Table 1 - ROC evaluation for the best accuracy achieved (Euclidean distances)

<table>
<thead>
<tr>
<th>Password</th>
<th>Threshold</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>mxplayer</td>
<td>460</td>
<td>76.4%</td>
</tr>
<tr>
<td>Lisboa2014</td>
<td>707</td>
<td>76.3%</td>
</tr>
<tr>
<td>tecnicoLisboa</td>
<td>363</td>
<td>77.1%</td>
</tr>
</tbody>
</table>

To be able to evaluate the performance of the system that uses Euclidean distances as a metric, and set an optimal threshold, the corresponding ROC curves were plotted – see Figure 6, Figure 7 and Figure 8. From Table 1, it is possible to see that using the password tecnicoLisboa leads to the best performance, while the other two passwords present similar results.

B. SVM

In machine learning SVM is a learning model with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. To be able to learn how to analyze data and recognize patterns, there has to be training data consisting of a set of training examples, each marked as belonging to one of two categories. Given that, an SVM training algorithm builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary classifier.

To be able to train and test the data parameter C and gamma had to be chosen for each set of users. The parameters where chosen based on the best output result. When training the data if the test user was the true user or a false user, the password being tested would not be in the training. This away the tested
password, does not correspond to any in training data. After some preliminary tests parameter C was set to the value 10000, for every password. Regarding parameter gamma, for mxplayer it was set to 1 and for Lisboa2014 and tecnicoLisboa it was set to 10.

To be able to evaluate the performance of the system when using the SVM classifier, and set an optimal threshold, the corresponding ROC curves were plotted – see Figure 9, Figure 10 and Figure 11. These curves also allow comparing the performance between the two approaches, using either the SVM classifier or the Euclidean distance metric.

In Table 2, the figures for the best accuracy achieved are represented. Despite the thresholds indicated in the table be an optimal solution, there are other possible thresholds than can be used depending on the goal of the application. If security is important the threshold should be higher and the opposite may happen if the user doesn’t mind if someone breaks in with his password.

Table 2- ROC evaluation for the best accuracy achieved (SVM)

<table>
<thead>
<tr>
<th>Password</th>
<th>Threshold</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>mxplayer</td>
<td>0.65</td>
<td>95.6%</td>
</tr>
<tr>
<td>Lisboa2014</td>
<td>0.39</td>
<td>97.8%</td>
</tr>
<tr>
<td>tecnicoLisboa</td>
<td>0.3</td>
<td>95%</td>
</tr>
</tbody>
</table>

The password Lisboa2014 has the best performance, while the other two have a similar performance. However, is clear to see that using the SVM classifier allowed achieving a big improvement over the results when using the Euclidean distance metric, as it is able to isolate each users’ characteristics much better, which is reflected in the classification and consequently on recognition results.

VI. CONCLUSION AND FURTHER WORK

In the last few years there have been some research in mobile keystroke dynamics. This work has been one more effort to enrich that research. However, with all the coding done by the author and the limited time involved, the application performance still has some room to be improved as well as the expansion of the database. So, it may be concluded that even with the limitation involving the application implementation, the application has a good performance and should allow increasing the security when entering alpha numeric passwords. Furthermore, by analyzing the system performance, notably looking at the ROC curves, it can be stated that using the SVM classifier the application can improve the password security for
a user. On the other hand, a system based on the Euclidean distance metric would still need further improvement before being applied to help improve security.

The author hopes that the developed app will be revisited in the future and that better performances may be achieved through the addition of new features. In this context, some features to enhance the algorithms classification are presented here:

- **Pressure**: adding the user finger pressure on the key may improve algorithm performance classification
- **Fixed weights**: adding fixed weights based on variance of the samples. A sample which is much more statistical dispersed naturally indicates a less reliable mean compared to a sample with a smaller variance, so variance is calculated in order to determine the statistical dispersion of the samples. Fixed weights would be assigned to each value in the template in order of the inverse ranking of their variance
- **Phone orientation**: analyze how the user holds its phone when training data through the gyroscope may help identifying the true user.

VII. REFERENCES


