Touchpad Input for Continuous Biometric Authentication

Alexander Chan[1], Tzipora Halevi[2], and Nasir Memon[2]

[1] Hunter College High School, New York, NY, USA
[2] New York University Polytechnic School of Engineering, Brooklyn, NY, USA

Abstract. Authentication is a process which is used for access control in computer security. However, common existing methods of authentication, which are based on authentication during the login stage, are insecure due to the lack of authentication after the initial instance. Ideally, authentication should be continuous and should not interfere with a user's normal behavior as to not create an inconvenience for the user. Behaviometric identification, for example, verifies a user's identity based on his behavior, both continuously and without interruption. This work shows that it is possible, with great accuracy, to identify different users based on their touchpad behaviors. While linear classifiers proved ineffective at classifying touchpad behavior, kernel density estimation and decision tree classification each proved capable of classifying data sets with over 90% accuracy.

Keywords: behavior, biometrics, behaviometrics, touchpad.

1 Introduction

In the context of digital security, authentication is a method of verifying identity. Authentication typically functions as a gatekeeper, granting certain users access to resources restricted to others. A web administrator, for example, a trusted user maintaining a website, can grant other users privileges and shut down the website. In order to grant the administrator access to such privileges while excluding all others, the administrator must verify his identity via authentication.

Authentication is dependent on three factors, which can be combined to increase security: ownership, knowledge, and inherence [2]. Ownership refers to a physical token that a user has, such as a credit card or a passport. Knowledge refers to something a user knows, such as a password. Inherence refers to features inherent to the user, or physical characteristics and behavior, for example, a fingerprint or typing rhythm.

In addition to these three factors of authentication, there are two main types of authentication, static and continuous. Static authentication methods verify a user's identity only once, at the first moment of access [1]. Once authenticated statically, however, any new user can subsequently obtain access to the original user's data simply by using the same device. Vulnerabilities such as these suggest that static authentication is inadequate for many situations in which security is a priority.

Continuous authentication minimizes this vulnerability. Unlike static methods, continuous authentications *continuously* verify a user's identity during session use, even long after the initial verification [1]. The simplest method for continuous authentication is repeated prompting for authentication (e.g. a password). However, this method is certain to inconvenience the user in proportion to the frequency of prompting or the level of security. Behaviometric identifiers, on the other hand, which monitor a user's behavior and authenticate the user by means of that behavior, solve this problem.

In this study we assess the usefulness of a touchpad, an input device found on many laptops, as a tool for behaviometric identification. It is found that the ability to authenticate a user based on touchpad data is achieved. Section 2 discusses related works, and our approach is presented in Section 3. Section 4 states our experimental results, and we conclude in Section 5.

2 Related Work

Behavioral biometrics have been widely explored with many different identifiers, including eye movements and pointing devices (mice) [4][5]. These authentication processes use pattern recognition. Gamboa et al. developed a preliminary biometric authentication system using two types of density estimation, multimodal non-parametric estimation and unimodal parametric estimation [4]. It was shown that there was no difference in the Equal Error Rate (EER), or the rate at which the false acceptance and rejection rates are equal, between the two algorithms. In addition, it was discovered that as the time the user used the pointing device to train the algorithm increased, the EER decreased.

In addition to the aforementioned examples of biometric identifiers, there have been numerous studies exploring touch sensors as a behavioral biometric, most commonly touchscreens [3][6][9][11]. Some of these studies focus on the act of drawing a "lock pattern" [1]. It was shown that each participant drew the lock pattern differently, and users could be differentiated. However, it was also shown that in order to obtain a low EER of 2%, at least 250 keystrokes were required. This time taken to train and build a classification model can be an inconvenience to users. Another disadvantage of authentication method is that it is static. It only analyzes initial login behavior.

Similar to this is biometric authentication using touch gestures on multi-touch devices. Sae-Bae et al. showed that touch gestures can uniquely identify users [9][10]. However, this study, like the one by Angulo et al., focused solely on static authentication.

Touch gestures as a means of continuous identification was explored by Frank et al. and Roy et al. [3][7]. Frank et al. analyzed smartphone scrolling behavior, classifying thirty different variables via k-nearest neighbor and support vector machine. Roy et al. used a Hidden Markov Model on mobile systems, improving on previous studies as to ease updating of trained classifiers, and as only the user of the device needs to provide training data. We expand on this approach by testing touchpad behavior, and comparing multiple algorithms to identify the most accurate and efficient one.

3 Data Collection

Data was collected exclusively from the built-in Multi-touch trackpad of a MacBook Air. The "MacBook Multitouch"[1] program was used to collect raw gesture data from each user's touchpad behavior for 15 minutes. Each participant "surfed the web."

Data from six different participants were collected. The participants were unaware that their behavior was being recorded, and so the program recorded their normal behavior. The data collected was stored in separate plain text log files. The structure of this raw data included a timestamp, an identifier for each individual finger, the finger size, the finger angle, the major and minor axis of the finger, the position, velocity and pressure of the finger, two relative position measures, and two different states. "States" refer to notable events such as lifting a finger or stopping finger movement. The relative position values indicated how far a finger was from the center or the edge of the touchpad.

A MATLAB script was written to format the raw data into a comma-separated value file. This file contained the same values as the raw data, except that data associated with distinct fingers was placed into distinct lists. Only the first 10,000 values were used from each file, and five participant's data were combined into a larger file. The data of the sixth participant was unable to be read by the software used, and was therefore omitted from the data analysis. This combined file also included new data values, the results of calculations performed on the original raw data. These calculations included the finger area, equal to the product of the major and minor axis; finger location, equal to the product of the x position and y position; absolute velocity $(v_x^2 + v_y^2)$; and the velocity angle $(\arctan \frac{v_x}{v_y})$.

The combined data file incorporated 12 features for each finger for five fingers, plus a value for the number of fingers on the touchpad at that instance, for a total of 61 features. Each individual participant's data was assigned a class name, a requirement for data set classification. Waikato Environment for Knowledge Analysis (WEKA) was used for data classification, a process in which algorithms build models based on training data to be able to predict the classification of future data points.
Six algorithms were tested in this work:

1. Simple logistic regression
2. Naive Bayes
3. Bayes network
4. J48
5. Random forest
6. k-nearest neighbor

[1] `http://www.steike.com/code/multitouch/`

Simple logistics, naive Bayes, and Bayes network are linear type classifiers. J48 and random forest both use decision trees as the primary classification technique. k-nearest neighbor, a kernel density estimation type classifier, was tested four different times, each time with a different k value.

For each trial, the model was built using a 10 fold cross validation. The data was partitioned into 10 sections. For each class, nine sections, or 9000 values, were used as training data, and the remaining section (1000 values) were used as test data. This was repeated ten times, such that each section was used as test data.

4 Results

The classification accuracy (identification rate), kappa statistic and relative absolute error were calculated using the number of correct classified instances by each classifier. The accuracy (identification rate) is simply the number of correctly classified instances over the total number of instances, in this case, 50,000. The kappa statistic takes into account chance agreement, and so is generally a more accurate indicator of how well a classifier performs than is the sample accuracy. The relative absolute error normalizes the total absolute error.

These values are listed in Table 1 for each classification algorithm.

Table 1. Performance accuracy of different algorithms

Classification Algorithm	Classification Accuracy	Kappa Statistic (κ)	Relative Absolute Error
Simple logistic regression	65.47%	0.5684	62.07%
Naive Bayes	29.65%	0.1207	87.53%
Bayes network	70.88%	0.6360	36.92%
1-nearest neighbor	95.13%	0.9391	6.10%
2-nearest neighbor	94.28%	0.9285	7.16%
3-nearest neighbor	94.14%	0.9267	8.14%
4-nearest neighbor	93.72%	0.9215	9.04%
Random forest	96.37%	0.9546	15.39%
J48	94.12%	0.9265	8.39%

Both the simple logistic regression and naive Bayes algorithms had a low sample accuracy when tested against the data set. Random Forest, J48, and k-nearest neighbor all performed well. The kappa statistic showed lower accuracy values for all classification algorithms, with a large reduction in accuracy for the naive Bayes, simple logistic regression, and Bayes network classifier. The relative absolute error values for the three poorest classifiers, simple logistic regression, naive Bayes, and Bayes network, were higher than the error rates for the other classifiers. Interestingly, the random forest classifier, although it had the highest sample accuracy and kappa statistic, also had a higher error rate than other similarly performing classifiers.

The k-nearest neighbor algorithm shows an interesting trend, namely, as the value of k increases, the sample accuracy and kappa statistic decrease, while the relative absolute error increases.

Confusion matrices were also generated for each algorithm. We have selected one confusion matrix representing each type of classification in Table 2-4.

Table 2. Naive Bayes

a	b	c	d	e	
205	444	269	2522	6560	a
44	365	132	4012	5447	b
62	75	1376	4847	3640	c
0	0	36	4040	5924	d
0	0	20	1140	8840	e

Table 3. k-nearest neighbor, $k = 1$

a	b	c	d	e	
9524	54	83	128	211	a
25	9790	83	45	57	b
160	145	9187	233	275	c
121	34	146	9348	351	d
117	24	32	111	9716	e

Table 4. Random forest

a	b	c	d	e	
9509	31	172	119	169	a
29	9821	100	35	15	b
119	75	9574	141	91	c
120	17	193	9551	119	d
59	13	80	119	9729	e

"a", "b", "c", "d", and "e" represent the different classes, or the users that generated test data. The top row represents the predicted class, and the side row represents the actual class. In the naive Bayes matrix, the majority of values were classified as either class "d" or class "e". Both k-nearest neighbor and random forest algorithms classified most values correctly.

5 Discussions and Conclusions

The low sample accuracy of the simple logistic regression and naive Bayes algorithms suggests that linear classification algorithms are not optimal methods to classify a data set of this nature. Since linear classification algorithms fail the data is not linear, which conclusion is in accord with that of other behavioral biometric studies [1]. Linear algorithms also took the longest time to generate a model based on test data.

The non-linear classifiers (decision tree and kernel density estimation) have a high sample accuracy and κ (both $> 90\%$). Random forest classification produced the highest sample accuracy, but it does not have the lowest relative absolute error, and it is not the fastest algorithm. k-NN (with $k = 1$) resulted in the lowest error rate.

It was reported by Angulo et al. that random forest classification resulted in the lowest error rate of the algorithms tested, and that performance was constant when testing various lock patterns [1]. Our study confirms these findings, as the different patterns tested can be compared to the wide ranges of behavior of a user on a touchpad. The various lock patterns that participants drew on the touchscreen can be compared to various touchpad gestures such as dragging a finger, two-finger scroll, and tapping. Random forest was also found to be the most accurate algorithm.

In a k-NN classification, the optimal value for k is 1. This is most likely due to other instances being too far away from the query point, expanding the nearest neighbors region, and thus lowering overall accuracy.

The confusion matrices reveal that the naive Bayes classifier predicted the majority of instances as either class "d" or class "e". This can be explained by looking at trends in the raw data. These two participants did not use more than 2 fingers at a time. Therefore, for the values of the other three fingers, zero was used as a placeholder. This made the model skew its predictions towards these two classes, and therefore most values were classified as either "d" or "e". One way to fix this problem is to ignore zero values, which may increase the accuracy of the naive Bayes classifier.

Future work includes using only one user's training data, similar to that described by Roy et al.. The classified data can also be modified to consist of entire strokes or gestures, rather than individual timestamps.

References

1. Angulo, J., Wästlund, E.: Exploring Touch-screen Biometrics for User Identification on Smart Phones. Privacy and Identity Management for Life 375, 130–143 (2012)
2. Campi, A.: How strong is strong user authentication? ISACA Journal 5, 42–45 (2012)
3. Frank, M., Biedert, R., Ma, E., Martinovic, I., Song, D.: Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. IEEE Transactions on Information Forensics & Security 8(1), 136–148 (2012), doi:10.1109/TIFS.2012.2225048
4. Gamboa, H., Fred, A.: A Behavioural Biometric System Based on Human Computer Interaction. In: Proc. SPIE, vol. 5404 (2004)
5. Juhola, M., Zhang, Y., Rasku, J.: Biometric verification of a subject through eye movements. Computers in Biology and Medicine 43, 42–50 (2013)
6. Kurkovsky, S., Syta, E.: Approaches and issues in location-aware continuous authentication. Computational Science and Engineering, 279–283 (2010), doi:10.1109/CSE.2010.42
7. Roy, A., Halevi, T., Memon, N.: An HMM-Based Behavior Modeling Approach for Continuous Mobile Authentication
8. Sae-Bae, N., Memon, N.: Online Signature Verification on Mobile Devices. 2014 IEEE Transactions on Information Forensics and Security 9(6), 933–947 (2014), doi:10.1109/TIFS
9. Sae-Bae, N., Ahmed, K., Isbister, K., Memon, N.: Biometric-Rich Gestures: A Novel Approach to Authentication on Multitouch Devices. In: Conference on Human Factors in Computing Systems (2012)
10. Sae-Bae, N., Memon, N., Isbister, K.: Investigating Multi-touch Gestures as a Novel Biometric Modality. In: 2012 IEEE Fifth International Conference on Biometrics: Theory, Applications and Systems, BTAS (2012)
11. Saevanee, H., Bhatarakosol, P.: User Authentication using Combination of Behavioral Biometrics over the Touchpad acting like Touch screen of Mobile Device. Computer and Electrical Engineering, 82–86 (2008), doi:10.1109/ICCEE.2008.157