Damian MAZUR [1], Mateusz TYBURA [2]
[1] DEPARTMENT OF ELECTRICAL AND COMPUTER FUNDAMENTALS,
[2] STUDENTS SCIENTIFIC CIRCLE OF INFORMATION TECHNOLOGY
RZESZOW UNIVERSITY OF TECHNOLOGY, 2 W. Pola St., 35-959 Rzeszow, Poland

Intelligent touch based user authentication

Abstract

Many researches had shown that touch based authentication is something possible to implement in many devices. This research focuses mainly on making a progress in this field by using more advanced methods such as SVM, kNN, kmeans or neural networks in attempt to build system for both recognizing and learning user's behavior.

Keywords: touch, mobile, authentication, authorization, SVM, kNN, kmeans, neural networks.

1. Introduction

Touching something is one of the natural ways of human communication. It lets us show our interests, relations or get some knowledge about environment. Nowadays more and more devices make it fully available to interact by touching its screen. On the other hand, it is not so useful in authorization process. It is just a method of putting data into particular device.

This research is an extension of previous work with more focus on this techniques and analysis of data on all steps of whole process. Hopefully it will make it all more reliable for usage on variety types of devices, including mobile phones, tablets or other with touch screens.

2. Available solutions

There are already some of approaches used in other researches. For instance, one is based on anthropometrics and uses that simple fact that swipe gesture would be different for users with different thumbs [2]. Other solutions propose to use flick gesture [3]. Those two seems to be too simple. Other example use much more sophisticated thing – surface electromyogram (s-EMG) [4], which unfortunately couldn't be used, when devices isn't capable of such a measurement. Of course, there have been many more "traditional" ones. They all tends to use keystrokes [5], as an input. It's quite good, but they are much more different than the ways that mobile phones give us to authenticate.

3. Previous approch

Method used in first research consisted of vectors and matrices as a representation of user's data, Euclidean distance as a measurement of pattern recognition ratio and randomly showing dots as an input.

This approach was working on some very small users "database", build of just 2 persons, but it's efficiency reduced for about 25% after increasing amount of data to 10 distinct people.

Since that it was decided to first analyze data gathered using this method and then figuring a way to reduce number of outliers and find other method of figuring out which user is interacting with device.

This other method should also take possible lack of floating point operations as a possible problem when implementing on some devices

4. First data analysis

First of all, we gathered 3 user's data as a material for analysis. It was inserted into CSV file and then processed using Octave into 6 histograms. One for each person's interactions property.

First user's reaction time (Fig. 1) seemed to be a little bit bell curved with a small outlier on the right site of plot. Also, there was a local peek in between very small values after global maximum and mentioned before outlier.

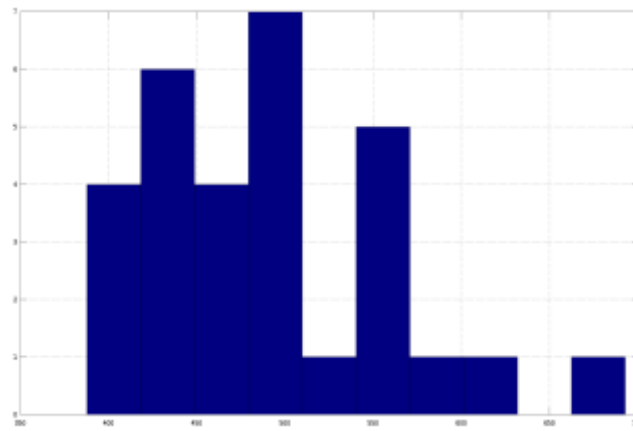

Fig. 1. First user's reaction time histogram

Accuracy (Fig. 2) also showed the same properties. But now outlier was placed on the left side. Again, there were few disturbances. First just before a global maximum, where value had rapidly fallen down. Second one on the right end of the plot, in which value had rapidly grown.

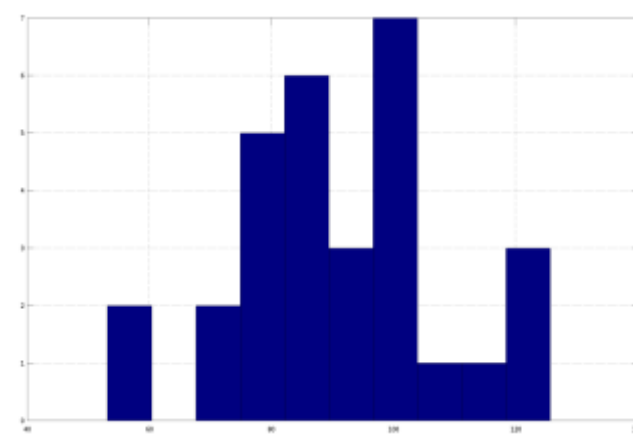

Fig. 2. First user's accuracy histogram

Next user's reaction time (Fig. 3) had been measured as more repetitive than the first ones. There were no significant changes in bell shape. On the other hand, one of the bars in plot is so far from most of the data, undouble an error in measurement.

Accuracy (Fig.4) instead covered almost whole values on the histogram, with only a little bit of space on both left and right side of the plot. There is only one peek which seems strange in neighborhood of smallest values on the right side of plot. Also, that smallest values slightly break the bell shape making.

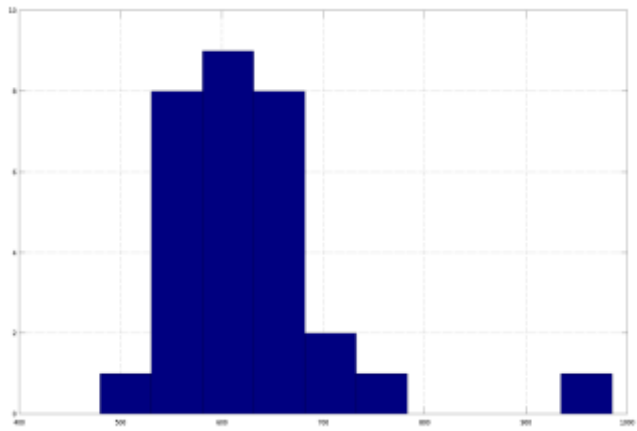

Fig. 3. Second user's reaction time histogram

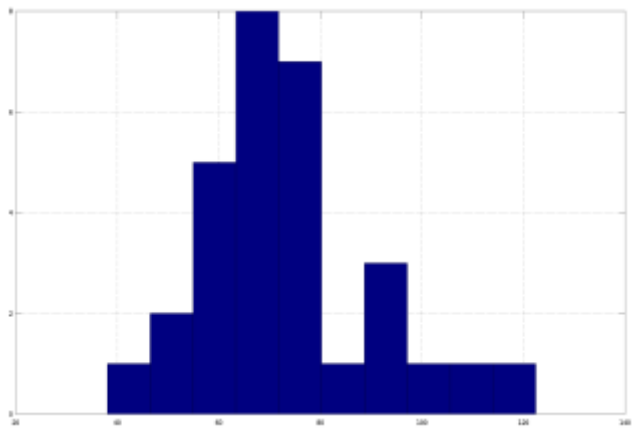

Fig. 4. Second user's accuracy histogram

Last user's reaction time (Fig. 5) have the same properties as the middle ones. Almost perfectly bell shape, which just one smallest outlier on the right side.

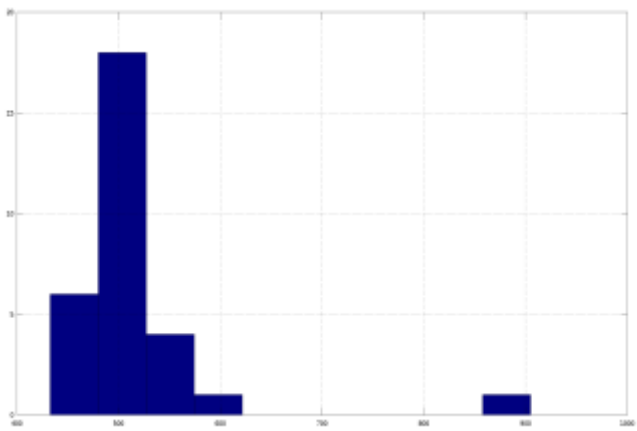

Fig. 5. Third user's reaction time histogram

Accuracy (Fig. 6) also showed good shape and only one single outlier.

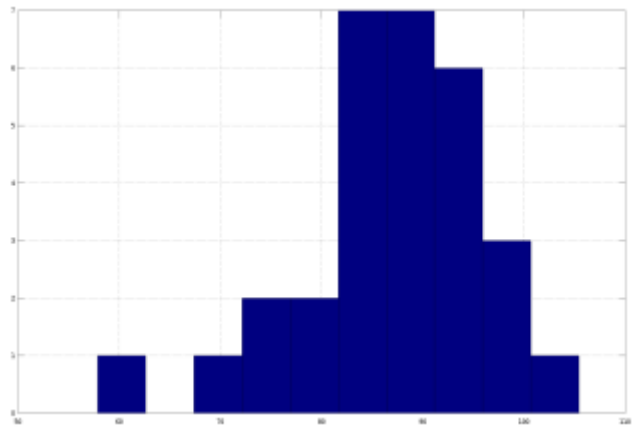

Fig. 6. Third user's accuracy histogram

All the other data was also fulfilling this observations about bell curved shape, outliers on the left or right side and sudden changes in particular points of the histograms.

As there were no certain value of outliers it wasn't fully possible just drop something below or over this value to prevent them or to filter them. The only reasonable way was to analyse users separately a than cutting of values which are suspected to be errors. It could be performed in spite of the fact that in all our data, unacceptable values seemed to be preceded by an empty space.

5. Method improvement

Firstly, randomness was reduced in order to gather more precise data. It is easy to prove that time of reaction (T_r), which is one of the measured properties, could be represented as a sum of realization time (t_r), movement time (t_m) and time of clicking (t_c).

$$T_r = t_r + t_m + t_c$$

Realization time is simply an amount of time which passed from the moment of appearance of dot till being noticed by user, as so it could be treated as a constant value of approximately 250 ms.

Next value, the time of movement, is a portion of time in which user was moving toward to dot. As the positions was generated randomly this certain value was considered as most variable.

Last value – time of clicking stands for number of milliseconds between stopping movement and clicking the dot.

Because of randomness in something which is not fully connected with user's characteristics, it was decided to make some serious change in way of presenting our login layer.

Previous way was changed into a grid of 9 dots showed all the time on a screen. This idea was divided into two separate one. First used two properties of user's characteristics and based on alteration of colour of randomly chosen dot in order to show which one must be clicked. Second one added two more properties and had extended interaction to not only clicking but also drag and drop interaction between central dot, and one that had been chosen by algorithm.

With two new ways of both interacting and gathering the data we decide to get data of 10 users.

6. Second data analysis

After reimplementing the methods of data acquisition, we tried again to get data and analyse them. Since there were more users, instead of visualizing as distinct histograms, all data was shown as colour dots and stars on plots.

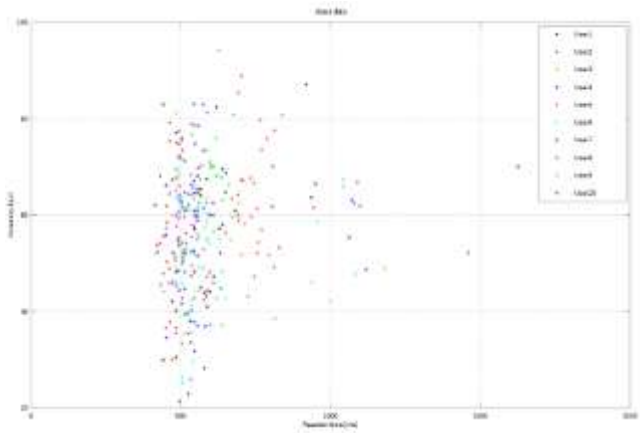

Fig. 7. User's data acquired using first method (reaction time – X axis, accuracy Y axis)

Data of 10 users (Fig. 7) acquired using first method showed large group of measured values to be from 500 to 1000 ms on reaction time axis, which almost all of them nearby left boundary of that particular set. It seems that accuracy is much more distinct, covering almost 75% of whole range of possible values.

There were also outliers. One absolutely clear on right side of plot, nearby legend. It is absolutely a value that has to be removed. Points between 1000 and 1500 ms could also be considered in that way. Together it will make 12 point to drop off.

Even much more distinct, the accuracy axis also has some values which seems to be a little bit different than the other ones. It seems that there could be a straight line separating them, which would have constant value around 82 on this axis.

Because second method consisted of 4 distinct values measured, plot was divided. First one (Fig. 8) presented reaction time and accuracy, and the second one (Fig. 9) showed interaction time and drop distance.

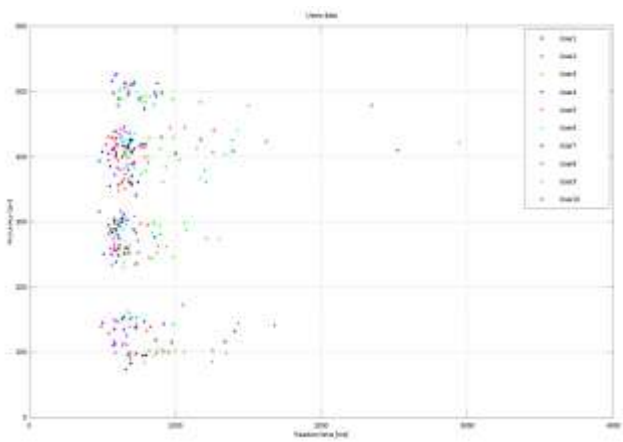

Fig. 8. User's data acquired using second method (reaction time – X axis, accuracy Y axis)

Almost all values of the reaction time were, as in previous method, between 500 and 1000 ms. But now there were much more values far from that range. Basing on this example it's rather good choice to remove all values bigger than 1000 ms.

Next value – accuracy has divided into 4 groups, instead of one big group as in first method. And this time there is no value considered as an outlier.

Last plot showed properties similar to the first one, but with values rotated 90 degrees clockwise. Drop distance has most of values between 200 and 300 px, with some outliers.

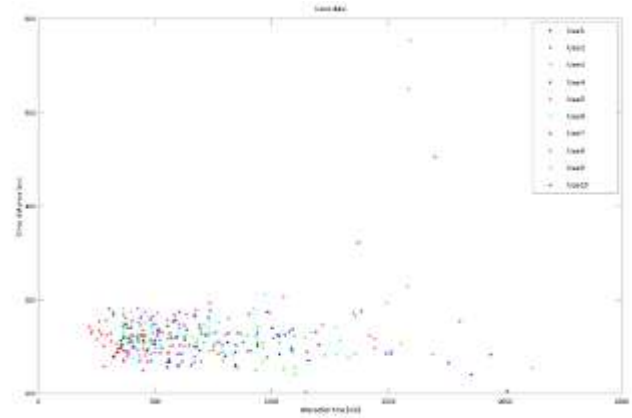

Fig. 9. User's data acquired using second method (interaction time – X axis, drop distance Y axis)

Interaction time set of values is a little bit bigger, than reaction time set. It now covers an area from about 250 to 1500 ms, making it bigger on the left and right side.

7. Classification

Since there was only one time calculation for building user model we decided to make it more useful by usage of classification methods. It is absolutely obvious that this method is a supervised one, as we know on the beginning which user inputs which data.

To choose one, wasn't so easy. Classificator must be good, which simply means that ideally there is no false positive or false negative. Also, it will be good if it could be calculated fast on every possible device with touch screen.

For a test, we chosen to use MATLAB environment because of possibility of fast building a prototype, which could be than implemented into our device. For checking more methods, restriction of fast calculation was omitted. After few considerations methods such as SVM, kNN, kmeans and neural networks were chosen. First one for the fact than in theory it gives us one, globally optimal separator. Next two were used in spite of fact that data seemed to take circular shapes on the 2d plot. Last one, the neural network for its ability of learning.

Unfortunately, every single method has failed for data collected in learning mode of our system. Support vectors not just failed, they don't even give any useful result, as the execution of script finished with error pointing that, there was request of allocation a 20.8GB array which exceeded maximum of array size. After few changes, it ended up with out of memory error. Because of this we dropped SVM from our tests.

Next, we used more simpler method – kNN. At the beginning, it is crucial to say that two possible parameter changes for *fitcknn* function were used. We selected 4 distance metrics: Euclidean, cityblock, Minkowski and Chebychev. For other one – number of neighbours there was a for loop starting in value of 1 and ending in value of size of user's data size. This method also failed, with error rate 65-76% of misclasifications (Tab. 1).

The third one failed as well. Even more there were no signification change in error, while changing distance parameter of *kmeans* function from sqeuclidean to cityblock, cosine and correlation. All approaches didn't work at all.

Tab. 1. Total errors for kNN method after training using different metrics and neighbors count

Metric	Value of k/ Total errors									
	1	3	5	7	9	11	13	15	17	19
Euclidean	69	73	65	66	72	72	69	73	72	72
cityblock	72	72	68	69	74	69	69	70	69	74
Minkowski	69	73	65	66	72	72	69	73	72	72
Chebychev	70	63	66	65	70	71	74	76	75	74

Finally, we had run a neural network. It was trained 36 times, as we used 12 train functions, and used them 3 times to test influence of random initial values. All of them had divided data to 3 groups. 75% into train data, 15% into validation data, and 15% into test data. Training ended up with values nearby 8 on mean squared error, which was completely unsatisfying (Tab. 2).

Tab. 2. Mean squared error of neural network after training using different training methods

Training function	Iteration	MSE
trainlm	1	7.968663
	2	8.14908
	3	7.668962
trainbr	1	8.237540
	2	8.250138
	3	8.248621
trainbfg	1	8.582926
	2	8.125015
	3	8.022676
trainrp	1	8.630761
	2	8.224388
	3	8.233520
trainscg	1	9.303814
	2	8.190188
	3	8.236573
traincgb	1	8.119635
	2	8.278259
	3	8.246448
traincgf	1	8.007262
	2	8.260097
	3	8.192108
traincgp	1	8.180767
	2	8.018977
	3	8.477190
trainoss	1	8.337426
	2	8.145450
	3	8.328760
traingdx	1	8.841231
	2	8.526475
	3	8.259220
traingdm	1	9.809494
	2	14.337752
	3	8.471278
traingd	1	9.243019
	2	8.321443
	3	8.124793

For this reason, we decided to preprocess data. It was done the same way as we presented data about users – using histogram. This simple process has given us distinction in all data. With simply grouping user rows into 5 rows, and then dividing all dimensions separately into 5 distinct measurements, we achieved success. Using kNN we found out that for 1 neighbor there is only 2 false recognitions for 20 test rows, when using our first measurement method. Even more there were no errors with 1 neighbor when using second of our measurement methods (Tab. 3).

Tab. 3. Total errors for kNN method after training on preprocessed data using different metrics and neighbors count

Metric	Value of k/ Total errors									
	1	3	5	7	9	11	13	15	17	19
Euclidean	0	13	14	16	17	17	17	17	17	17
cityblock	0	13	14	16	16	18	16	17	17	13
Minkowski	0	13	14	16	17	17	17	17	17	17
Chebychev	0	15	16	17	17	18	18	16	15	16

8. Conclusions

Identifying users by how they interact with touch screens is possible. Even more it seems that it could be done using simple methods such as kNN, and with only 4 types of values measured.

On the other hand, there is still place for researches on things like distributing user's data between devices, filtering the data, making system fully able to learn and even more. Certainly, all of these things would be taken into account in our future researches focused on this kind of user authorization process. Interaction time set of values is a little bit bigger, than reaction time set. It now covers an area from about 250 to 1500 ms, making it bigger on the left and right side.

9. References

[1] Tybura M., Szczepański A.: Touch screen based user identification, MAM, 12/2015.

[2] Bevan C., Fraser D. S.: Different strokes for different folks? Revealing the physical characteristics of smartphone users from their swipe gesture, International Journal of Human-Computer Studies, vol. 88, 2016, pp. 51-61.

[3] Lin C., Chang C., Liang D.: A novel Nonintrusive User Authentication Method Based on Touch Gestures for Smartphones, Journal Of Internet Technology, vol. 16, issue 5, 2015, pp. 801-810.

[4] Yamaba H., Nagatomo S., Aburada K., Kubota S., Katayama T., Park M., Okazaki N.: An Authentication Method Independent of Tap Operation on the Touchscreen of a Mobile Device, Journal of Robotics Networking and Artifical Life, vol. 2, issue 1, 2015, pp. 60-63.

[5] Liu C. L., Tsai C. J., Chang T. Y., Tsai W. J., Zhong P. K.: Implementing multiple biometric features for a recall-based graphical keystroke dynamics authentication system on smart phone, Journal of Network and Compture Applications, vol. 53, 2015, pp. 128-139.

[6] Boyce J., Shapiro J. R. and Tidrow R.: Windows 8.1 Bible, ISBN 13: 9781118835319, Wiley, 2014.

Received: 02.10.2016 *Paper reviewed* *Accepted: 02.12.2016*

Damian MAZUR, DSc, PhD

Damian Mazur works at The Faculty of Electrical and Computer Engineering in Rzeszow University of Technology as a lecturer. At his didactic work he is dealing with diagnostics of the electromechanical devices (calculations and measurements of the electric machines), numerical methods (finite element method, boundary element method), object programming and databases.

e-mail: mazur@prz.edu.pl

Mateusz TYBURA, Msc, eng.

Mateusz Tybura has graduated from Rzeszow University of Technology in Computer Science (Msc, eng). Currently he is both working as an software developer and studying for PhD degree on the same university. He is also a member of KNEiTI scientific circle. His main interests are security, mobile technologies and artificial intelligence.

e-mail: tyburam@hotmail.com