CIS4930/CIS6930

Biometric Authentication on Mobile Devices

Projects and Reports

University of South Florida

Dept. of Computer Science and Engineering

Dr. Tempestt Neal, Instructor

Effective Touch Dynamics Features

Abstract- The purpose of this research is to determine which touch dynamics features would be more optimal to use from the list of all possible features in touch dynamics. By finding the most efficient features to use, researchers can then use this information to create a better model from this biometric. The goal of this paper is to examine the procedure that was taken to retrieve the optimal and efficient features from the list of features in touch dynamics. Given the list of optimal features, the glimpse of using these features will allow the study of mobile biometrics to consider using the touch dynamics biometric modality more.

I. Introduction

As number of sensors in a mobile device increase and the performance of these sensors in mobile devices improve, the more optimal the data gathered from these mobiles will be. The need of finding optimal data from these sensors are needed to improve the uniqueness of each mobile device from another one. This will allow the mobile device to be more secure and this security feature will help remove the threat of gathering data from a mobile device.

Of the many biometrics modalities that have been gathered from mobile devices, touch dynamics is an interesting modality that is used constantly by users of mobile devices. The data that is captured by this biometric is however very costly for algorithms to process, has a high energy consumption for the battery of the device, this leads to low accuracy rates in finding imposter and genuine scores and also has a factor of user adaptation to their touch dynamics which must be factored in when using this biometric.

With these current challenges, the data that is collected from touch dynamics has a large amount features which have could have been a reason for all the problems that is faced with this biometric. Perhaps, finding the best of all the features that is collected would help alleviate the challenges and allow for this biometric to be more optimal for future use.

In this paper, the overview of touch dynamics and the challenges of this biometric is discussed in Section 2. In Sections 3 and 4, the features of touch dynamics will be examined as well as the process of how to consider effective features. Lastly, Section 4 and 5 will discuss the results of the implementation and the datasets which were used in the process and the conclusion of the research.

II. Touch Dynamics

Of the many biometric modalities, touch dynamics, which is the study of measuring and assessing user interaction with a touch screen device, is considered promising in the field of biometric authentication as the main interaction of the touch screen is constantly used as the user interacts with the device. This biometric allows a user to not require any physically body parts such as the iris or face to gather information needed for the results of the biometric.

In the early conception of this biometric, was the use of keystrokes from a computer keyboard to gather information of the user habits when typing. The birth of

keystroke dynamics was considered, after the usage of its information was proposed to be useful in the field of authentication.

As modern devices have used less keyboard input for user interaction, the rise of touch screens have made progress to take over the mobile device landscape. The amount of data this is gathered from the touch screens can as well be used for biometric authentication as the same features of keystroke dynamics are present in the touch screen. Even more features were captured when a user interacts, however the amount of features have led to a surplus of data that may or may not be effective to gather.

The challenges of touch dynamics are:

- Reducing energy consumption - Mobile phones, unlike computers, run on batteries and hence power supply is limited. Therefore, the less the application takes memories, the more the device can be operated. So it is essential to have a system that consumes minimum battery so that it does not majorly impact the phone usage time. Some measures, such as reducing the sampling rate (Niu and Chen, 2012) or performing complex computation only when a device is being recharged (Crawford et al., 2013)
- Reducing computational cost- Computational cost in mobile phones are generally inferior than computer devices. Hence criterions such as criteria such as algorithm complexity, communication cost, and authentication delay are important and should be considered in the design of touch dynamics authentication solutions. In other words, algorithm and communication costs introduced as the result of deploying this authentication means should be minimal.(Teh and Zhang, 2016)
- Adaptation Capability: Human behavioral characteristics are susceptible to change over time, and more frequently than physiological characteristics. A user's touch dynamics patterns can gradually change as the user gets more familiar with the passcode, input method, device, and other external factors. A touch dynamics authentication system should be capable of adapting itself to any changes in a user's touch dynamics pattern.(Teh and Zhang, 2016)

Throughout this paper we try to create a model which addresses these issues.

III. Features of Touch Dynamics

We used two dataset in our experiments. The first was created by Dr. Neil Tempest on a Nexus 7. The dataset contained 71 features which were all numerical along with 2856 records. The measured attributes are:

- Hold (H)
- Up-Down (UD)
- Down-Down (DD)
- Pressure (P)
- Finger-Area (A)
- Averages of Hold (AH)
- Pressure (AP)
- Finger Area (AFA)

The second dataset used was the MOBIKEY Keystroke Dynamics Password Database. We limited ourselves to the evoline1 subsection of it. The dataset had 14 features and ~15,000 records. One of the

columns was the UserID i.e. the label we intend to predict using KNN in this case.

IV. Implementation of the process

Our implementation was divided into feature selection and the decision making. We employed the use of a Python library, scikit-learn, to perform the following types of feature selection algorithms:

- *Chi square*, this statistical algorithm that measures the frequency of expected and observed data, was used to filter the amount of features of touch dynamics. This implementation was done by getting the p-value of the features which value was less than a given alpha value.
- *Information gain*, using entropy in this algorithm we were able to also filter out the best features which had a high information gain.
- *Recursive*, for this algorithm, our goal was to select features by recursively considering smaller and smaller sets of features. The estimator was defined as a support vector machine Then, the least important features were removed from current set of features.That procedure was recursively repeated on the removed set until the desired number of features to select is eventually reached.
- *Tree selection*, in this algorithm, we used tree based estimators. It employs the process of tree selection to determine feature importance. It then uses the average of the importance to remove unneeded features
- *Variance threshold*, in this algorithm, we remove all features whose variance doesn't meet the declared threshold. By default, the code removed all zero-variance features, i.e. features that have the same value in all samples. We used the formula for variance V= p(1-p), to achieve this.

The script breaks up the dataset using 5 fold cross validation. The chunk meant to be used for testing was left alone, and the rest had outliers removed. Outlier detection was done using isolation forest in combination with leave-one-out-validation. Thus every row was tested as being an outlier against every other row. In practice we found that only a small percentage of rows were dropped; about 5%. The remaining training rows post outlier detection were used for the KNN training. Also, the training and query chunks are scaled using a MinMaxScaler after outlier detection. We chose k=31 since that yielded passable results during debugging with a partial database, but later tested other values for k.

Lastly, the main script was adapted into another script called a validation script. The purpose of this was to test our method on another database - the second database. This would serve to confirm or reject out method. The script initially used all the same code with only minor changes to account for different location of the dataset and type of file to read. The script, however, did not work. More specifically the method used for the first dataset did not work when applied to the second dataset. Essentially this inherently rejects our method, however, we went further in our attempts to make it work. First we noticed that there were simply too much data, ~15,000 rows vs ~2,500 for the first dataset. Out script never finished even when given several hours. To remedy this we switched to taking a random sample of 2,000 rows. Secondly, we noticed the data had text data and our script was meant to work with

numerical. To remedy this we applied a label encoder. Thirdly, we noticed that the recursive feature elimination algorithm took far too long. To remedy this we changed the SVM estimator for a Logistic Estimator that would perform much faster. However, the estimator would not converge thus the dataset can not be fit to a logistic curve thus a different estimator was needed. Then, an SGD Estimator was used. It worked considerably faster and did converge. Next we hit a problem we could not solve: the feature selection algorithms did not agree on a set of features to keep after the intersection of them were taken. For the machine learning algorithm to work two or more feature columns were needed. We saw that after 5 attempts to get selected features at best 1 feature was chosen by all 5 algorithms.

V. Results

The first set of results are the ones required when using only a small portion of the first dataset. We used ~350 rows of the first dataset when building our script. When the script was done we observed that with k for KNN being 31 for 12 runs we averaged 65.7% accuracy. The number of features chosen were between 3 - 6, collectively they were

- Hold i
- Hold Shift
- Hold Caps
- Hold a
- Hold n
- Hold l

The next set of results come from testing on the entire dataset. Below are the ROC, DET, and Score distributions along with the D-prime value and the EER value. The accuracy for these results, were very low, strangely low, for integrity they were 5.6% to 7%. With the 7% being reached with k=75 and the lowest coming from k=81.

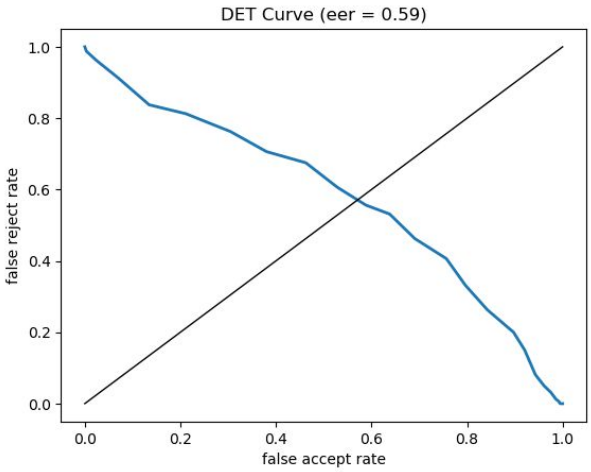


Figure 1. The DET curve with the EER value.

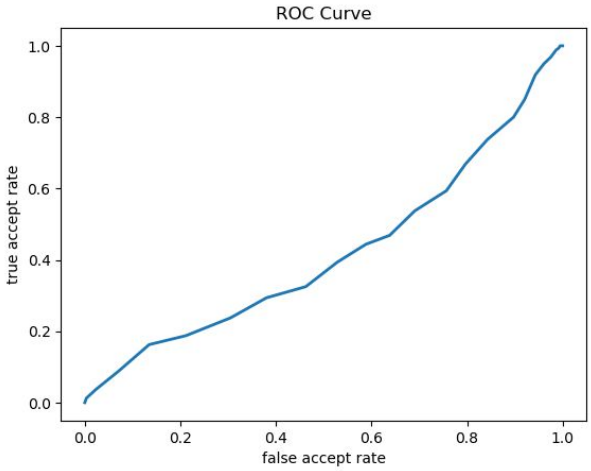


Figure 2. The ROC curve with the D-Prime value

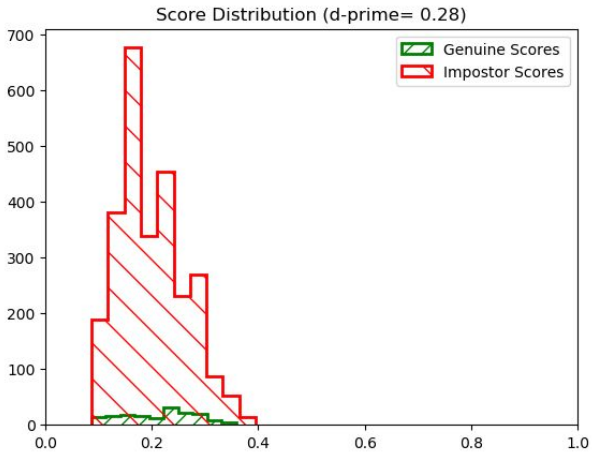


Figure 3. The Score distribution between genuine and imposter scores

There is one last set of results to make note of. How the script performs when using the entirety of the dataset and every feature. As mentioned before, with k=81 we got 5.6% accuracy for KNN, but using all the features with k=81 we got 1.9% to 2%. That shows a significant drop relative to the performance with feature selection. This points to KNN itself being the likely culprit for failure.

VI. Conclusion

Our approach to feature selection yielded fairly high results when used on small samples. Even though our results were, far from ideal when it came to testing on the entire dataset, we determined the major problem existed in the speed of performing all the necessary calculation. Our approach of using k Nearest Neighbor, seemed to be the issue, as our k value was not ideal for our dataset. We were unable to determine a good k value as the runtime of our entire code exceeded thirty minutes, and hence multiple tests on sample datasets were not feasible. We determined some sort of tree approach like the random forest would have been better for future works. This would ensure us to achieve our goal of achieving a system, which would seamlessly implement the touch gesture as a feasible mode of biometric authentication.

VII. Sources

- Teh, Pin Shen, et al. "A survey on touch dynamics authentication in mobile devices." *Computers & Security* 59 (2016): 210-235..
- Niu, Yuan, and Hao Chen. "Gesture authentication with touch input for mobile devices." *International Conference on Security and Privacy in Mobile Information and Communication Systems*. Springer, Berlin, Heidelberg, 2011.
- Jain, Anil K., Arun A. Ross, and Karthik Nandakumar. *Introduction to biometrics*. Springer Science & Business Media, 2011.

Password Strength and Keystroke Dynamics of Mobile Devices

Introduction

Security is something that is important to everyone. A lot of people struggle to determine what is a good password to have for their devices. With the data that provided to us, we looked at three variations of increasing difficulty passwords (easy: kicsikutyatarka, logical strong: Kktsf2!2014, strong: .tie5Roan). The data gave us multiple keystroke information, the ones that we look at were Down Time the time of a key press, Uptime, the time of a key release and Pressure the measurement of the finger pressure on the screen. With all of this information that was given, we wanted to see if adding capital letters, numbers and punctuation to the password would make the user hesitate, or add more pressure while typing the password.

Background

Traditionally, the primary methods for securing our mobile devices have been knowledge-based authentication, such as PINs or passwords. Due to security vulnerabilities associated with knowledge-based authentication, research in biometric authentications methods has proven a viable and more secure authentication method in our mobile devices. The two types of biometric authentication techniques are physiological and behavior. Physiological measures user features such as fingerprints, facial recognition, and iris recognition. Behavior measures things like gait patterns, keystroke dynamics, voice recognition, and signature recognition.

Some of our initial thoughts for our research topic were derived from articles and surveys like [1], [2]. The survey [2], looks more directly at the relatively new behavioral authentication technique of keystroke dynamics on modern touchscreen mobile devices. Intrigued, we decided to dig deeper into the potential for this authentication method. S. Karatzouni and N. Clarke explain, “keystroke analysis combines features that can offer a cost-effective, non-intrusive and continuous authentication solution for mobile devices” [3]. In this article, they attempt to verify an individual’s identity by looking at their typing patterns. The two patterns they observed were inter-keystroke latency and hold-time. Inter-keystroke latency is the interval between two successive keystrokes. Hold-time is the interval between the pressing and releasing of a single key. The use of keystroke authentication that really looked interesting was found in the idea of Password Hardening [4]. In this article, the proposal was to strengthen the traditional knowledge-based authentication method by introducing the advantages of keystroke dynamics into the mix. A hardened password is a password that

must match passcode value in addition to the typing patterns (i.e. latency between keystrokes, and duration of keystrokes). This method also includes the users typing pattern automatically adapt to the user over time making for a more secure password scheme.
In our research, we wanted to determine if the difficulty level of the user's password has a direct effect on the keystroke dynamics and habits.

Method

The Modality that we choose to work with was keystroke, we look at three specific area of the data, Down Time the time of a key press, Uptime, the time of a key release and Pressure the measurement of the finger pressure on the screen. We wanted to see if making a password more difficult to type by adding capital letters, numbers and punctuation would make the user change any of their patterns compared to the other set of passwords. Our approach to the problem was quite simple we calculated the average of Uptime, Downtime, and pressure for each user and for the overall for each password and looked if there was a difference between all of the given passwords. Since we are looking at only three features for this project, we decided to do feature selection to see if any of the data would stand out enough to consider. The feature selection that we ran was to remove any features that were above a certain threshold, but once we ran the code the feature selection was unsuccessful to remove any other features, the reason why the mean-variance that we achieved was such a high number that nothing from the data excited that number.

Result

Looking at the data without any calculation we believed that as the user typed the more complex passwords, that the user would have hesitated, and applied more pressure for each keystroke the user did.

Figure 1: Overall average

```
Overall average for each of the passwords
----------------------Easy----------------
        Preassure: 6.113827740157703e-07

        UpTime: 26.426655297589143

        DownTime: 26.426294411933284

-------------------Logical Strong---------
        Preassure: 6.6233135609099O8e-07

        UpTime: 17.382959828473172

        DownTime: 17.38250341426135

---------------Strong--------------
        Preassure: 4.2457137670225364e-07

        UpTime: 17.196732215767167

        DownTime: 17.196343025353986
```

Once we ran the calculations it was the complete opposite as shown in figure 1 that shows the overall average for pressure, up-time, and downtime for each of the passwords, we can observe that for the password that is considered easy the averages are relatively high, from easy to logical strong we notice that pressure goes up by .5, uptime and downtime get reduce by 45%, from logical strong to strong we notice that up-time and downtime is decreased slightly, but pressure goes down substantially. We also calculated the delay of the user between key press for each of the passwords to see if the difference in time between all of the passwords for each user, shown in figure 3.

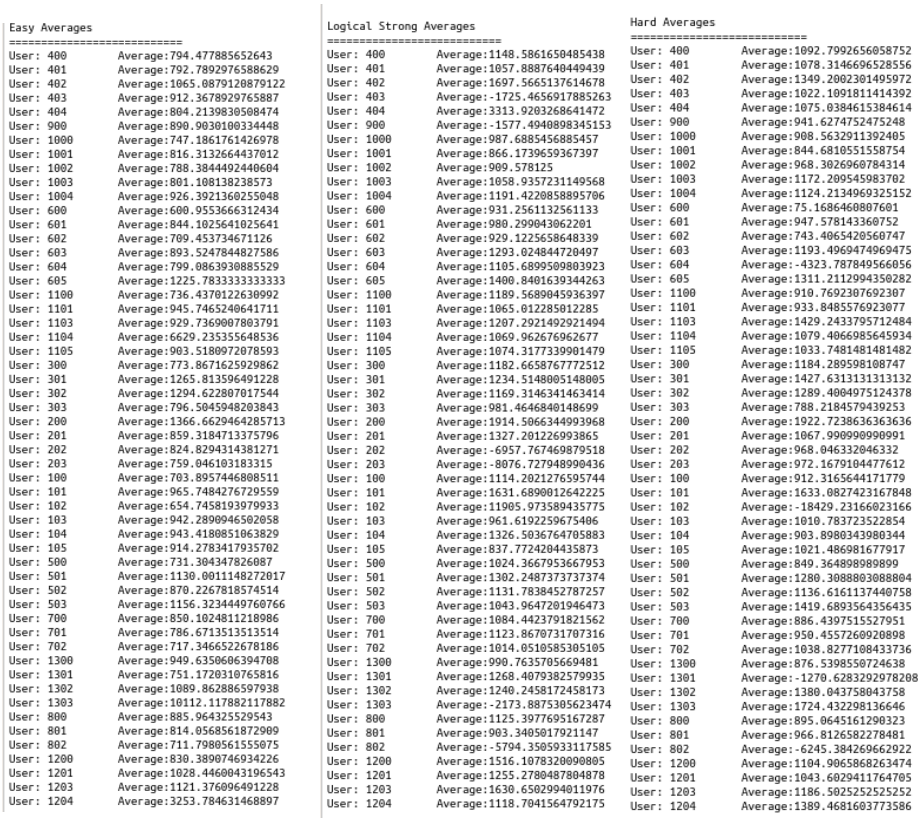
Figure 2: Averages of 10 different users

```
        Preassure,              Uptime,           DownTime
[[1.4935823346519693e-05, 138.21917541812525, 138.2024504084014],
 [1.9603268685669l6e-05, 132.21411901983663, 132.195254764683],
 [2.2403735019779543e-05, 21.180280046674447, 21.15908206923376]]


[[2.1044606590743652e-05, 217.7859321387724, 217.77011056042699],
 [3.0194436531234143e-05, 346.9064048799085, 346.8917270301182],
 [3.110941782099483e-05, 355.92241707967975, 355.9067861227602]]


[[8.621632657168387e-05, 345.63666536509174, 345.6126513080828],
 [4.498243087532464e-05, 25.38442014837954, 25.358453729012105],
 [2.7176884411369066e-05, 434.8971105037095, 434.8717297930496]]


[[1.164106765073832e-05, 159.10852869846403, 159.08407437348424],
 [2.037186895345813e-05, 247.82497978981408, 247.8110347615198],
 [2.037186895345813e-05, 255.94523039611965, 255.92784963621665]]


[[3.165244489700134e-05, 164.0178432893716, 164.00620636152055],
 [3.537626126369272e-05, 159.65748642358417, 159.64740108611326],
 [1.3964314265154978e-05, 99.93502715283165, 99.9245539177657]]


[[3.0298395944912013e-05, 1258.3754781943383, 1258.364001530222],
 [2.2035196059678653e-05, 1244.0570007651108, 1244.0390206579955],
 [2.4789595065034434e-05, 1264.2595638867635, 1264.2461744452946]]


[[4.007767126213592e-05, 265.24970873786407, 265.2378640776699],
 [3.262136174757281e-05, 368.65728155339804, 368.6423300970874],
 [3.355339883495145e-05, 374.9102912621359, 374.8935922330097]]


[[2.6373628357328836e-05, 75.22704081632654, 75.19917582417582],
 [5.0863423611343e-05, 167.19701726844585, 167.17268445839875],
 [2.8257458536553793e-05, 118.03963893249608, 118.01844583987442]]


[[3.665521223156358e-05, 62.27109583810615, 62.24474990454372],
 [4.215349605767133e-05, 42.74532264222986, 42.71954944635357],
 [3.3906073163328576e-05, 236.34956090110728, 236.331996945399]]
```

These outcomes were not what we expected, we looked deeper at individual averages and saw the same pattern shown in figure 2 . From what we can see we notice the same pattern that we got from the overall average, from this we are able to conclude that when this experiment was conducted, the users were told to type the first password multiple times and then the second and finally the third. After gathering all of the data we can conclude that while the users did struggle and took their time to type the easy password, once they began to type

the logical strong and strong passwords, the users were already been accustomed to using the keyboard on the phone, and do not struggle or hesitate to type the other password

Figure 3: delay between presses of same password type



Easy Averages
==========
User: 400 Average:794.477885652643
User: 401 Average:792.7892976588629
User: 402 Average:1065.0879120879122
User: 403 Average:912.3678929765887
User: 404 Average:804.2139830508474
User: 900 Average:890.9030100334448
User: 1000 Average:747.1861761426978
User: 1001 Average:816.3132664437012
User: 1002 Average:788.3844492440604
User: 1003 Average:801.108138238573
User: 1004 Average:926.3921360255048
User: 600 Average:600.9553666312434
User: 601 Average:844.1025641025641
User: 602 Average:709.453734671126
User: 603 Average:893.5247844827586
User: 604 Average:799.0863930885529
User: 605 Average:1225.7833333333333
User: 1100 Average:736.4370122630992
User: 1101 Average:945.7465240641711
User: 1103 Average:929.7369007803791
User: 1104 Average:6629.235355648536
User: 1105 Average:903.5180972078593
User: 300 Average:773.8671625929862
User: 301 Average:1265.813596491228
User: 302 Average:1294.622807017544
User: 303 Average:796.5045948203843
User: 200 Average:1366.6629464285713
User: 201 Average:859.3184713375796
User: 202 Average:824.8294314381271
User: 203 Average:759.046103183315
User: 100 Average:703.8957446808511
User: 101 Average:965.7484276729559
User: 102 Average:654.7458193979933
User: 103 Average:942.2890946502058
User: 104 Average:943.4180851063829
User: 105 Average:914.2783417935702
User: 500 Average:731.304347826087
User: 501 Average:1130.0011148272017
User: 502 Average:870.2267818574514
User: 503 Average:1156.3234449760766
User: 700 Average:850.1024811218986
User: 701 Average:786.6713513513514
User: 702 Average:717.3466522678186
User: 1300 Average:949.6350606394708
User: 1301 Average:751.1720310765816
User: 1302 Average:1089.862886597938
User: 1303 Average:10112.117882117882
User: 800 Average:885.964325529543
User: 801 Average:814.0568561872909
User: 802 Average:711.7980561555075
User: 1200 Average:830.3890746934226
User: 1201 Average:1028.4460043196543
User: 1203 Average:1121.376096491228
User: 1204 Average:3253.784631468897

Logical Strong Averages
==========
User: 400 Average:1148.5861650485438
User: 401 Average:1057.8887640449439
User: 402 Average:1697.5665137614678
User: 403 Average:-1725.4656917885263
User: 404 Average:3313.9203268641472
User: 900 Average:-1577.4940898345153
User: 1000 Average:987.6885456885457
User: 1001 Average:866.1739659367397
User: 1002 Average:909.578125
User: 1003 Average:1058.9357231149568
User: 1004 Average:1191.4220858895706
User: 600 Average:931.2561132561133
User: 601 Average:980.299043062201
User: 602 Average:929.1225658648339
User: 603 Average:1293.024844720497
User: 604 Average:1105.6899509803923
User: 605 Average:1400.8401639344263
User: 1100 Average:1189.5689045936397
User: 1101 Average:1065.012285012285
User: 1103 Average:1207.2921492921494
User: 1104 Average:1069.962676962677
User: 1105 Average:1074.3177339901479
User: 300 Average:1182.6658767772512
User: 301 Average:1234.5148005148005
User: 302 Average:1169.3146341463414
User: 303 Average:981.4646840148699
User: 200 Average:1914.5066344993968
User: 201 Average:1327.201226993865
User: 202 Average:-6957.767469879518
User: 203 Average:-8076.727948990436
User: 100 Average:1114.2021276595744
User: 101 Average:1631.6890012642225
User: 102 Average:11905.973589435775
User: 103 Average:961.6192259675406
User: 104 Average:1326.5036764705883
User: 105 Average:837.7724204435873
User: 500 Average:1024.3667953667953
User: 501 Average:1302.2487373737374
User: 502 Average:1131.7838452787257
User: 503 Average:1043.9647201946473
User: 700 Average:1084.4423791821562
User: 701 Average:1123.8670731707316
User: 702 Average:1014.0510585305105
User: 1300 Average:990.7635705669481
User: 1301 Average:1268.4079382579935
User: 1302 Average:1240.2458172458173
User: 1303 Average:-2173.8875305623474
User: 800 Average:1125.3977695167287
User: 801 Average:903.3405017921147
User: 802 Average:-5794.3505933117585
User: 1200 Average:1516.1078320090805
User: 1201 Average:1255.2780487804878
User: 1203 Average:1630.6502994011976
User: 1204 Average:1118.7041564792175

Hard Averages
==========
User: 400 Average:1092.7992656058752
User: 401 Average:1078.3146696528556
User: 402 Average:1349.2002301495972
User: 403 Average:1022.1091811414392
User: 404 Average:1075.0384615384614
User: 900 Average:941.6274752475248
User: 1000 Average:908.5632911392405
User: 1001 Average:844.6810551558754
User: 1002 Average:968.3026960784314
User: 1003 Average:1172.209545983702
User: 1004 Average:1124.2134969325152
User: 600 Average:75.1686460807601
User: 601 Average:947.578143360752
User: 602 Average:743.4065420560747
User: 603 Average:1193.4969474969475
User: 604 Average:-4323.787849566056
User: 605 Average:1311.2112994350282
User: 1100 Average:910.7692307692307
User: 1101 Average:933.8485576923077
User: 1103 Average:1429.2433795712484
User: 1104 Average:1079.4066985645934
User: 1105 Average:1033.7481481481482
User: 300 Average:1184.289598108747
User: 301 Average:1427.6313131313132
User: 302 Average:1289.4004975124378
User: 303 Average:788.2184579439253
User: 200 Average:1922.7238636363636
User: 201 Average:1067.990990990991
User: 202 Average:968.046332046332
User: 203 Average:972.1679104477612
User: 100 Average:912.3165644171779
User: 101 Average:1633.0827423167848
User: 102 Average:-18429.23166023166
User: 103 Average:1010.783723522854
User: 104 Average:903.8980343980344
User: 105 Average:1021.486981677917
User: 500 Average:849.364898989899
User: 501 Average:1280.3088803088804
User: 502 Average:1136.6161137440758
User: 503 Average:1419.6893564356435
User: 700 Average:886.4397515527951
User: 701 Average:950.4557260920898
User: 702 Average:1038.8277108433736
User: 1300 Average:876.5398550724638
User: 1301 Average:-1270.6283292978208
User: 1302 Average:1380.043758043758
User: 1303 Average:1724.432298136646
User: 800 Average:895.0645161290323
User: 801 Average:966.8126582278481
User: 802 Average:-6245.384269662922
User: 1200 Average:1104.90658682634
User: 1201 Average:1043.6029411764705
User: 1203 Average:1186.5025252525252
User: 1204 Average:1389.4681603773586

Summary

Understanding the importance of protecting our mobile devices and the information we keep in them is paramount in this ever-advancing digital world. Understanding that the most common level of protection people use to protect their devices is knowledge-based authentication (passwords, pins, or patterns) and the potential threat they pose to attacks seemed a cause for concern. We found an interesting article by F. Monrose et al. [4] suggesting the idea of password hardening, which involves combining the traditional knowledge-based authentication with a keystroke dynamic. This idea results in a non-intrusive two-step authentication process where the user must input the correct password with a correct set of keystroke patterns. We decided to look a little deeper into this idea and determine if the complexity of the user's password had any effect on the keystroke patterns. The three features we looked at were: Down Time the time of a key press, Uptime the time of a key release and Pressure the measurement of the finger pressure on the screen.

The results that we received from this project were not the ones that we expected. We believe that the reason we got the numbers shown is that, the users typed the passwords continuously one after another making the users accustomed to typing in the keyboard of the phone, and that is something that we did put into consideration while performing this project. Another feature that could be introduced to this project that would have added deeper insight between users keystroke and password difficulty would have been if the user types the passwords with one or two hands and see the comparison of the selected features.

Something that could have been done differently in this project would have to do with the data collection. If we were to reevaluate this hypothesis it may serve us better in the future to collect our own data to better apply to our research.

References

[1]W. Meng, D. S. Wong, S. Furnell, and J. Zhou, “Surveying the Development of Biometric User Authentication on Mobile Phones,” *An introduction to biometric recognition - IEEE Journals & Magazine*, 2015. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7000543&isnumber=7214344&tag=1. [Accessed: 20-Nov-2018].

[2] P. S. Teh, N. Zhang, A. B. J. Teoh, and K. Chen, “A survey on touch dynamics authentication in mobile devices,” *NeuroImage*, 18-Mar-2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404816300256?via=ihub. [Accessed: 20-Nov-2018].

[3]Karatzouni S., Clarke N. (2007) Keystroke Analysis for Thumb-based Keyboards on Mobile Devices. In: Venter H., Eloff M., Labuschagne L., Eloff J., von Solms R. (eds) New Approaches for Security, Privacy and Trust in Complex Environments. SEC 2007. IFIP International Federation for Information Processing, vol 232. Springer, Boston, MA

[4]Monrose, F., Reiter, M. & Wetzel, S. IJIS (2002) 1: 69. https://doi.org/10.1007/s102070100006

Keystroke Dynamics for Identification

Introduction:

There have been many research done on keystroke dynamics biometrics on cellular devices. Ever since the introduction of cell phones, there have been studies conducted in user authentication for higher security in both conventional and non-conventional mobiles phones. During one of these thefts, data stored on the phone could contain sensitive information of the user for further misuse. To combat this to an extent, knowledge-based authentications were introduced such as passwords. But they can still be susceptible to attacks from an imposter.

Because these threats are still prominent with knowledge-based authentication, we must consider the use of biometric authentication. Biometric authentication is an approach used to identify a person's identity using his or her behavioral or physiological traits. Behavioral traits considered during a biometric identification includes signatures, gait or keystroke dynamics. Keystroke dynamics refers to the automated method of identifying or confirming an individual based on the rhythm of typing on a keyboard (Keystroke Dynamics, n.d.). When it comes to biometric user authentication, there must be a fair balance between usability and security.

The objective of this reports is to present a background on keystroke dynamics by reviewing published works that uses keystroke authentication on both conventional and non-conventional mobile devices. Go through our approach to user based biometric authentication. Present the results of our experiments. The final section will be a discussion of our findings and results.

Background:

Touch/Keystroke dynamics biometrics refers to the process of measuring and assessing human touch rhythm on touchscreen mobile devices, a form of digital signals generated by the interaction of humans and these devices (Teh, Zhang, Teoh, Chen, 2016). These digital signals that are being read will be used for classifying a genuine user form an imposter when one is trying to access the device. Keystroke recognition is not limited to conventional devices, meaning mobile devices that involve touch screen, but to non-conventional devices such as flip phones, keyboards, etc. as well. Using keystroke dynamics is not a state of the art form of authentication. But since most mobile devices released during present day tend to be conventional, our project also focuses on datasets that originate from touch screen mobile phones.

A survey on touch dynamics done by Teh et el. (2016) summarizes the usefulness and user-friendly requirements of a keystroke dynamics and its attached challenges. When it comes to the authentication processes, a high authentication frequency lowers the usability of the user. This means that a system can't be too secure and have many security bypass process as a user will probably not have the time nor patient to go through them every time they access their devices. Combining both knowledge based, and biometric authentication will provide a good balance of security and usability. Touch dynamics can generate multidimensional features such as timing, and motion features that can be hard to replicate forming distinctiveness. A combination of passcodes and touch dynamics will increase the security. Keystroke is less sensitive

to environmental factors making it ideal for mobile devices. When comparing to other biometric authentication methods, touch dynamics is far more cost effective due to the mobile devices already having inbuilt hardware. But touch/keystroke dynamics comes with its fair share of challenges.

Because the capabilities of a mobile device are lower than a computer, algorithm and communication cost must be minimal. So, there is no room for any complex algorithms. Some of the specialized in-built hardware used for touch dynamics have a direct impact on costing the battery life of the device. When working with touch dynamics, the accuracy performance is relatively low compared to other biometric methods. Data extraction can vary between the user's use of the device at different times and external factors such as mood or injuries can affect it. Similarly, the user's touch patterns can gradually change as the user gets familiar with the input process, so an authentication system should be able to adapt to the change of the user's pattern.

Method:

Our primary objective of our project is to implement a keystroke authentication method and discuss the results of our findings. The dataset that we used to identify genuine users and imposters comes from the Sapientia University. This dataset is public and is available for download on the university's website. As mentioned before, the dataset consists of keystroke dynamics that was collected on the Nexus 7, a touch screen mobile. The dataset contains 2856 records, 51 records per subject which there are 56 of them. During the data collection process, the subject was asked to type the string '.tie5Roan1' as their keystroke information was collected. Each measured attribute includes Hold (H), Up-Down (UD), Down-Down (DD), Pressure (P), Finger-Area (A), Averages of Hold (AH), Pressure (AP) and Finger Area (AFA). There are 71 features, 41 for timing feature and 10 for touch screen features because each feature has a set of feature elements that correspond to the typed character.

The main method that our group implemented for this project was calculating the minimum Euclidean distance from a query to a template. The outline for the project is as follows: Read all data files into the project and partition the data into training and test data (we chose 46 of the 51 samples for training and tested on 5). We calculated different weights for each feature based on the variance throughout a feature. The variance of each feature column was calculated for each user. If a feature has very low variance, then it is not a good metric for identification. The higher the variance, the better the feature would be for identification. We defined an inverse relationship where Weight = k + c / var. K and c were constants that we adjusted as an independent variable to observe how the different weighting would affect our results. Essentially, the more something weighed, the more important that feature was for identification. Each specific Subject had a set of weights that was unique to them, with the strongest features for identifying them having the heaviest weights.

After establishing the weights, we then went through the process of calculating the Euclidean distance by comparing each Subject to every other Subject. This was done by computing the sum of the distances from each template feature to each query feature. The weights were applied by multiplying both of these values by the weight before computing the difference. As each Subject has a unique weight set, the distance from the query and imposter templates should be exaggerated and greater. In theory,

the lowest distance should be the genuine attempt, with all imposter attempts greater than it.

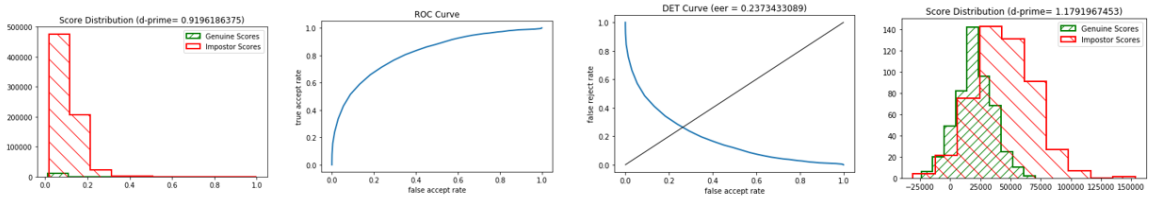
The secondary method we implemented was by using feature selection. After we established our weights, we ran a function to calculate the top 10 weights from the weights array that we stored the values to. Since each weight and its indices indicate what feature they belong to, we can use the top 10 weights to trim our dataset down to top 10 features. So, our new feature selected data set will consist of 56 subjects, 51 records and 10 features. Once we collect our new data, we can use Euclidean distance to find the genuine and imposter score distribution.

Results:

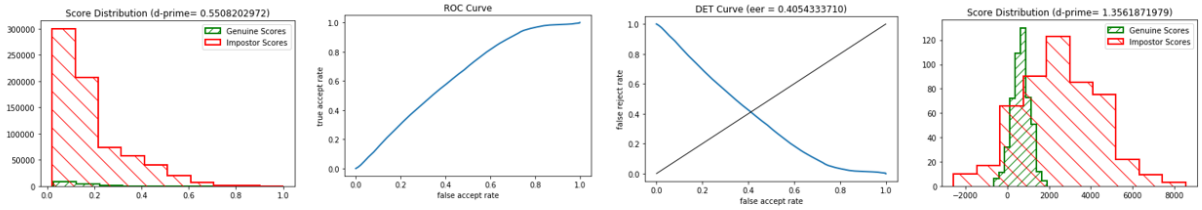
(I) Euclidean Distance

For clarity, included in the graphs will be a generated graph of fabricated data that is characteristic of our imposter and genuine data. That is, the data will have the same mean and standard deviation but will be generated on a histogram with the same number of samples, so it is graphically easier to read.

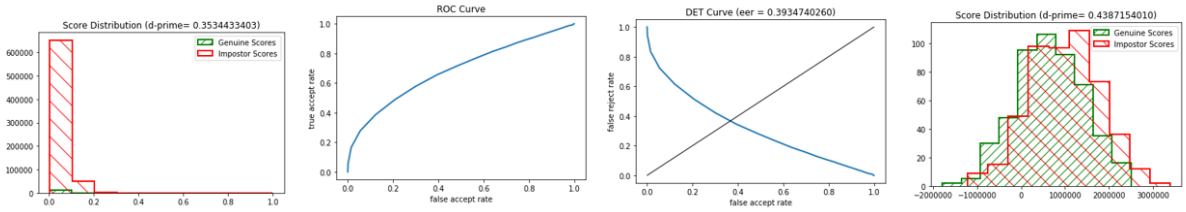
1. Weight function: Weight = 2 + 10 / Variance (Best Function)
   D-prime: 1.18 EER: 23.7%



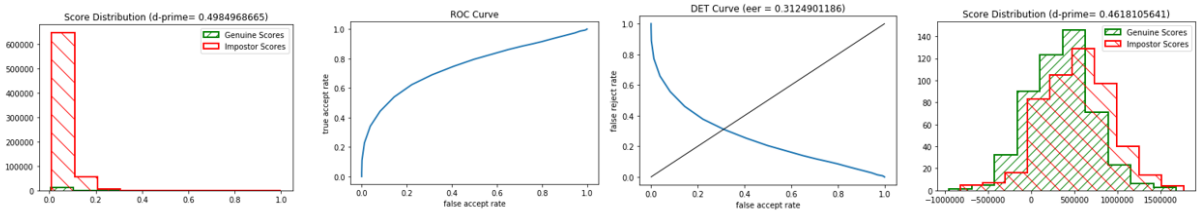
2) Weight function: Weight = 1 / Variance
   D-prime: 0.55 EER: 40%



3) Weight function: Weight = 100 + 1 / Variance
   D-prime: 0.35 EER: 39%



4) Weight function: Weight = 50 + 30 / Variance
D-Prime: 0.50 EER: 31%

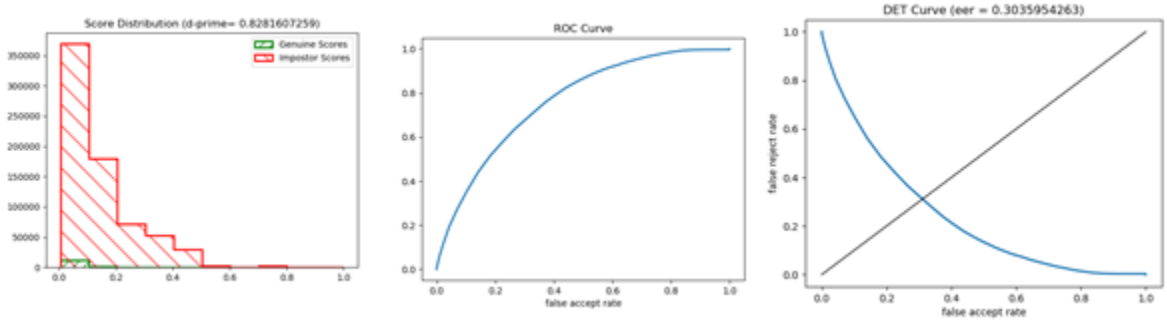


After seeing many different score distributions, it is obvious why we chose to fabricate characteristic results. After much trial and error, it was decided that the first function listed was the best for selecting weights.

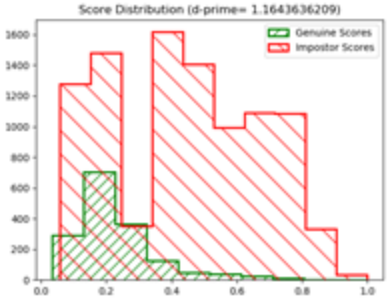
(II) Feature Selection:

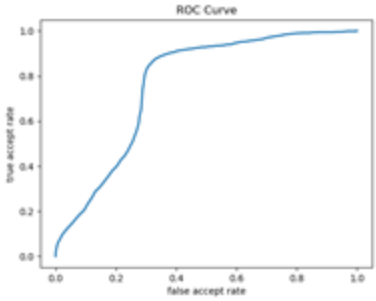
We ran feature selection on seven different sets of data. The first 6 were just the 56 subjects divided by 10, while the other consisted of all 56 of the subjects. The purpose of this little experiment was to find a specific reason for our results when training and testing with the 56 subjects. To compare the data from a small set to a large one. The D-prime values for the different datasets are: 1-10: 0.85; 11-20: 0.74; 21-30: 0.31; 31-40: 0.96; 40-50: 0.77; 50-56: 1.16; 1-56: 0.86. The data sets from 50-56 produced the best d-prime score compared to the others. So, this could mean that there are outliers in datasets from 1 to 49 that are bringing each d-prime value down. See below for a graphical comparison between datasets 50-56 and from 1-50:

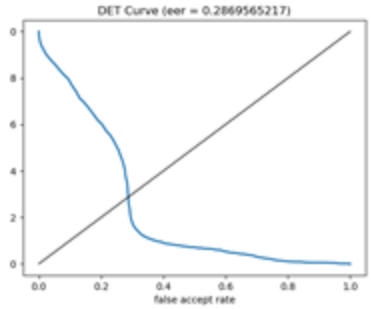
1) 1 to 56 subjects with feature selection results:



2) 50 to 56 subjects with feature selection results:







Based on results we learned that making the training data will always make the testing run more accurate. For our experiments we split our datasets by 46 for training and the remaining for testing. We tried changing the dividing number to 20 or 15 and we got varying results that was of course lower than our 46 splits. Another important feature that we should not look past are the weights. Weights are used for noise reduction, that will help our training data when it comes to keystroke features. Data enhancement maybe needed for other biometrics but think that noise reduction is very useful for keystroke features.

Discussion:

Keystroke dynamics is a flawed but effective concept. As the demand for smartphones continue to increase the demand for better security will as well. Keystroke dynamics has great value when considering cost-efficiency and a higher security. But it also comes with its challenges. One of the challenges stated by Teh et el. (2016), was the usage pattern of the user will change gradually. We ran into this concept when we were working with our dataset. The Sapientia University dataset records a user typing in a password for 51 continuous days. So, the input pattern for the password will gradually differ from the initial pattern skewing our results. This could be one of the possible explanation to our results.

One big take away we got from working on this project was how important it is to train on a big data. During our initial testing process, we were splitting the training data and the testing data evenly. This makes it difficult for the program to distinguish between a genuine and an imposter when there is an outlier in the dataset. This outlier maybe someone who might've injured their fingers before typing in the password. The recording of the inputs might've varied drastically from all the other recording. So, when taking this outlier into account during our training process will skew the score distribution when it comes to the testing. This was prevalent when we were splitting the training and testing data evenly. However, we can account for this by have more data to train on. So, if the classifier does eventually run into another outlier during the testing phase, it will ignore it completely.

For our experiments we worked with noise reduction, feature selection and weight changes. But one technique that we would like to implement in the future would be Naïve Bayes classification. Maybe our results would've been completely different or the same if we would've used machine learning. Using Euclidean distance is an effective way of classification, but our best d-prime value was 1.18 when we changed the weight calculation. We did have a low EER in all our experiments, but we felt as if machine learning would've fared better than using distance classification. Especially

techniques such as random forest learning when working with large dataset. For any future work with keystroke dynamics we would like to experiment on the machine learning end.

References:

- Teh, Pin Shen, et al. “A Survey on Touch Dynamics Authentication in Mobile Devices.” *Computers & Security*, vol. 59, 2016, pp. 210–235., doi:10.1016/j.cose.2016.03.003.
- Al-Obaidi, Noor Mahmood, and Mudhafar M. Al-Jarrah. “Statistical Median-Based Classifier Model for Keystroke Dynamics on Mobile Devices.” *2016 Sixth International Conference on Digital Information Processing and Communications (ICDIPC)*, 2016, doi:10.1109/icdipc.2016.7470816.
- Admin. “Keystroke Dynamics.” *Biometrics*, www.biometric-solutions.com/keystroke-dynamics.html.

Using PCA to evaluate the impact of data size and threshold on Gait Recognition

Biometrics for Mobile Authentication Course Project
Department of Computer Science
University of South Florida
4202 E. Fowler Avenue, Tampa, FL 33620, USA

Abstract

This paper presents a novel approach to gait recognition using data enhancement, Principal Component Analysis (PCA), independent match threshold based on standard deviation, and variable training and validation datasets which attempts to provide further insight into how these variables impact the performance of the gait recognition system.

Introduction

Gait recognition is the systematic analysis of a person's walking style through the use of video recording, floor sensors, and wearable sensors. It can be used in a wide variety of applications. For example, in medical diagnostics, pathological gait can be responsible for the causation of a symptom in conditions such as cerebral palsy and strokes.[1] The study of gait allows diagnoses and intervention strategies to be made, as well as permitting future developments in rehabilitation engineering. Gait can also be used in professional sports training to optimize and improve athletic performance. It also has chiropractic and osteopathic utilizations in terms of detecting misaligned pelvises and sacrums.[1] Once this is detected, doctors can better discern the listing of a pelvis and accurately restore a full range of motion to areas involved in ambulatory movement. Gait is also used in comparative biometrics by studying the gaits of non-human animals, yielding more insight about the mechanics of locomotion.[1] This information can be applied to better understand the biology of the species in question, as well as locomotion more broadly.

Gait recognition is important because it can reveal underlying information that is commonly overlooked by most people. A person's ability to walk is the most basic method of transportation, however the inability to walk can have drastic changes on a person's life. Many people can move with abnormal gait patterns for years without any symptoms, however when an injury is experienced and that gait is altered, more serious health issues can occur. These can include musculoskeletal problems from altering movements to compensate for pain or discomfort, cardiovascular health issues due to inactivity, and mental health issues from depression and loss of independence.[2] This is why the study of gait recognition is so important, as it identifies an individual's unique movements, determines normal gait patterns, diagnoses issues causing pain, and implements and evaluates treatments to correct abnormalities.

We will discuss our experiment design and findings regarding data collected from "Personalization and user verification in wearable system using biometric walking patterns" [Casale, Pujol, and Radeva]. Our design description will include our usage of Principal Component Analysis (PCA) in order to enhance the obtained data, followed by

the incorporation of weighted moving average in order to reduce noise and enhance gait cycle data. We will also go over the layout of our algorithm step-by-step, and the results we gathered from testing several training data sets against generated validation sets in order to identify whether or not a particular user is known to the system. Furthermore, we will show the impact that changing the margin of error rate has over time when matching different users training data to validation sets.

Background

In a study of biometric recognition by gait, it was discovered that there are seven stages of a gait cycle: loading response, mid-stance, terminal stance, pre-swing, initial swing, mid-swing, and terminal swing.[3] Gait was also found to be measured via Ground Reaction Forces between the pressure and angle of the foot contacting the ground and rising. Audio was determined as an aspect that can measure gait by recording the sound of footsteps in order to extract mel-frequency, cadence, and power-spectral.[3] It was discovered that a challenge of using gait as a biometric mobile device was that the device's orientation can shift while in motion or be carried in a different area on a person, which has an impact on the results obtained. Also, deep learning can be a useful tool for gait recognition, as it can extract features from multiple modalities instead of having to manually select them.

Another study was conducted for identifying users of portable devices from gait pattern accelerometers. The experiment involved a user wearing a portable device on their belt, which measured accelerometer signals that were gathered on a laptop. The user walked 20 meters at three different paces: normal, fast, and slow. This process was redone after five days to get a second data set, which was compared against the first set for genuine and imposter values. The main objective was to determine the effects of analyzing walking speed in a biometric device that can accept or reject a valid user in a system. It was discovered that gait biometrics should not be the sole user authentication method for portable devices, but rather as a complementary method for existing ones.[4] Also, changes in the speed of a person's walk, type of shoes worn, type of ground walked on, degree of sobriety, injuries, positioning and placement of the accelerometer device on the person has a major impact on the variation of collected data.

Method

Our approach to the problem includes many techniques we learned throughout the course, including: varying training sample size, data enhancement via weighted moving average, Principal Component Analysis(PCA), and iterating with different threshold values to determine the ideal threshold value for the system.

Our independent variables were training sample size and the threshold for a match. We initially started with a training sample size of 2 cycles per user, and 2 validation sets. We then increased these values up to 16 training sample cycles per user, and 4 validation samples per user. The combination of these numbers were limited due to one user only have 20 detected cycles.

Our experimental design can be broken down into two main steps. First, we detect cycles from the gathered data and split them into X training data sets (one for

each cycle), and Y validation sets. (Values X and Y were varied to measure the impact it had on the test results) Then, we store these sets in a retrievable way for later when we check whether each validation set is correctly or incorrectly matched to each user. The first step involves generating the templates for each user and storing them for matching and decision in later use. Data extraction and feature enhancement is performed for each training data set, and the modified versions will be saved. The second step involves validating the system, in which feature extraction and data enhancement is performed on each validation set. This process follows the identification method as a particular user's data will be checked against every set to see if it does not match. Matching and decision-making will be used for comparison to the training data and determining match success rate. Score distribution plots are generated, and the margin of error is changed for each validation set. We used gait as our biometric modality. Gait is considered a behavioral modality because it changes over time.

Feature selection was used in both of the main steps. When the templates were generated in the first step, PCA was performed on each of the training data sets. When the system was being validated in the second step, the PCA feature was used and dot product was performed on each of the validation sets. A matcher was used in the second step to compare each validation set's features to each training data set. A variable threshold was used for decision making which initially started at five percent to return if there are successful matches, else change in value and try matching again.

Results

Cycle detection seems to work well, however when testing the system as a whole, we had a low match rate with the validation data set. Conversely, for the training data set, we observed an approximately 70% match rate for configurations. An unexpected set of results, was the number of Principal Component Analysis (PCA) components retained had little to no impact on the accuracy of the matching or Equal Error Rate (EER).

PCA,Train/Val, σ	EER, %Correct
1, 10/10, 1	0.51, 5%
1, 10/10, 5	0.26, 5%
1, 15/5, 1	0.51, 6%
1,15/5, 5	0.23, 6%
1, 18/2, 1	0.36, 7%
1, 18/2, 5	0.14, 7%

PCA,Train/Val, σ	EER, %Correct
2, 10/10, 1	0.50, 5%
2, 10/10, 5	0.26, 5%
2, 15/5, 1	0.50, 7%
2,15/5, 5	0.19, 7%
2, 18/2, 1	0.55, 5%
2, 18/2, 5	0.55, 6%

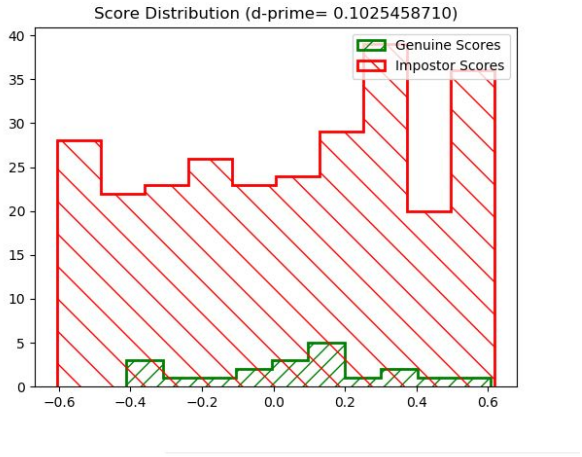
Overall there was an increase in accuracy as amount of training data increased. However the EER fluctuated with no apparent pattern in relation to increasing validation

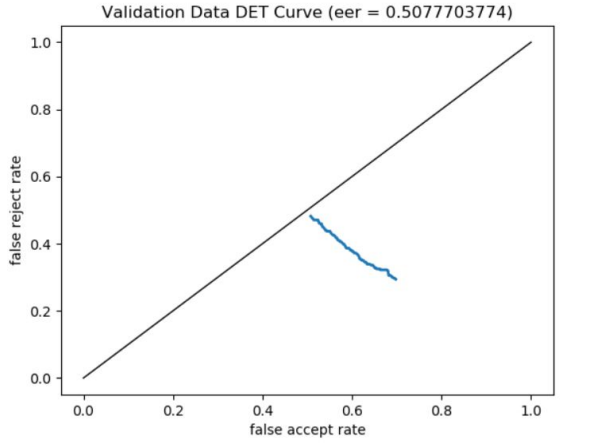
success. The effects of specific variables on the total result are somewhat obfuscated due to PCA's function in reducing this data into sub data. Finally, increasing the standard deviation seemed to improve EER in some cases, but not other experiment configurations.

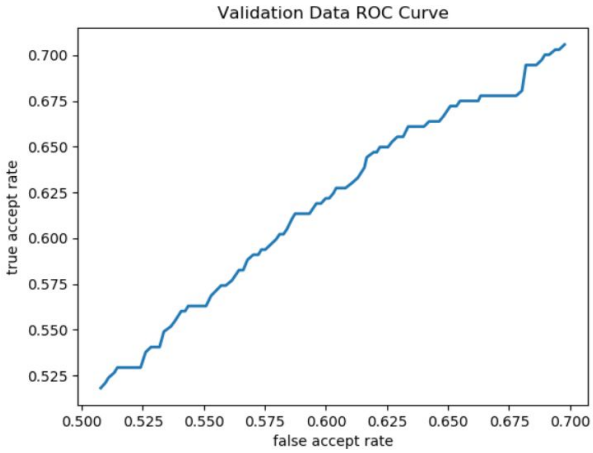
Charts

Experiments 1 and 2 and their related charts below shows that while the data changed slightly from 1 PCA component kept to 2 PCA components kept, all other variables stayed constant. This had very little impact on the ERR of the system.

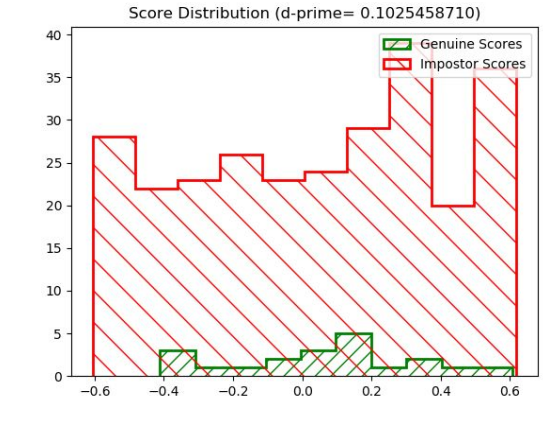
Experiment 1:1 PCA components kept, 10 Training Data Samples, 10 Validation Samples, 1 Threshold Multiplier

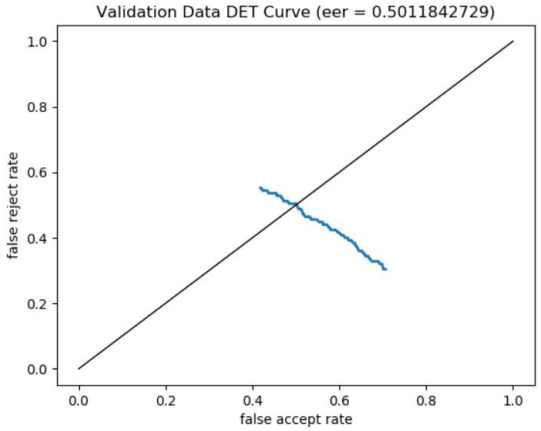


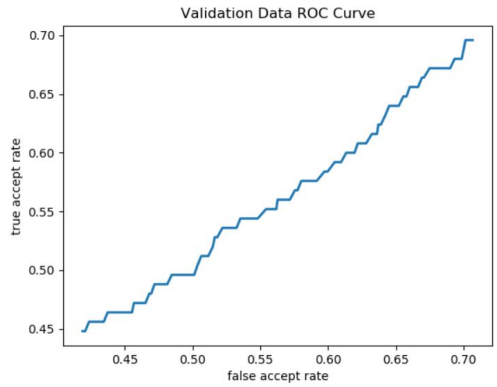




Experiment 2: 2 PCA components kept, 10 Training Data Samples, 10 Validation Samples, 1 Threshold Multiplier

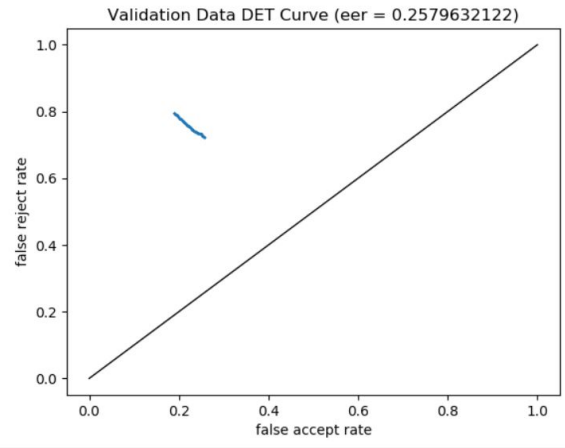


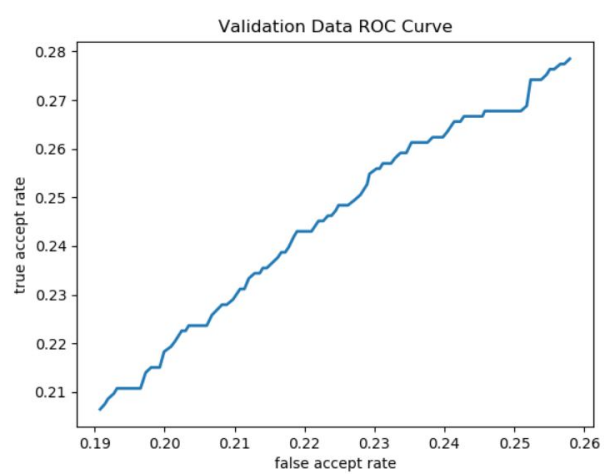




In Experiment 3, below, we see that increasing the threshold multiplier decreases the EER of the system. Compare these results to the experiments above, as the all other variables are the same.

Experiment 3: 1 PCA components kept, 10 Training Data Samples, 10 Validation Samples, 5 Threshold Multiplier





Summary

Gait is a unique and highly personalized aspect of an individual's biometric data that can be implemented in a variety of ways. Aside from allowing access to mobile devices and secure areas, gait recognition can be used in medical diagnostics to detect physical and mental changes, as well as in biological practices to distinguish locomotive characteristics from different species and develop biomechanic technology. Studies have shown that detecting a person's gait cycle to be used as a biometric modality requires several methods of enhancing data gathered from experiments, since a number of factors can contribute noise and inaccuracies that negatively affect it. Principal Component Analysis and Weighted Moving Average were two methods that we found have caused fluctuations in the results. However, despite changes in techniques, there were not many substantive increases in accuracy. Fluctuations at times showed better EERs or increased accuracy. Despite this, they leveled off and did not offer much increase in overall success.

In our experiment we learned that increasing the acceptance threshold, which was the standard deviation in this case, led to a decrease in the Equal Error Rate. This alteration however, did not improve the matching percentage. This relationship wasn't always true, for some experiments, the EER did not improve with increased threshold. Furthermore, we learned that the number of PCA components kept had little impact on the EER of the system. Similar to prior studies, we have discovered that there exists underlying data characteristics that can create limitations while designing a particular biometric system, which is why it is recommended to test more than one biometric in a system to ensure more accurate data retrieval and generate more personalized user templates.

A few factors to consider in the failures of this experiment might include use of PCA and the technical implementation. Further review of the implementation may reveal inherent flaws such as methodology in verifying a query. Specifically, around the matcher functionality. Additionally, testing one verification data sample against all users may inherently create biases in success statistics.

References

1. “Gait Analysis.” *Wikipedia*, Wikimedia Foundation, 8 Nov. 2018, en.wikipedia.org/wiki/Gait_analysis.

2. “Why Is Gait Analysis Important?” *Tekscan*, 2 Jan. 2018, www.tekscan.com/blog/medical/why-gait-analysis-important.

3. Connor, Patrick, and Arun Ross. "Biometric recognition by gait: A survey of modalities and features." Computer Vision and Image Understanding 167 (2018): 1-27.
URL: https://doi.org/10.1016/j.cviu.2018.01.007

4. J. Mantyjarvi, M. Lindholm, E. Vildjiounaite, S. -M. Makela, H. A. Ailisto. “Identifying Users of Portable Devices from Gait Pattern with Accelerometers - IEEE Conference Publication.” *An Introduction to Biometric Recognition - IEEE Journals & Magazine*, ieeexplore.ieee.org/abstract/document/1415569.

5. P. Casale, O. Pujol, P. Radeva. “Personalization and User Verification in Wearable Systems Using Biometric Walking Patterns.” *Contents: Using the Digital Library*, ACM, dl.acm.org/citation.cfm?id=2339117.

6. Gafurov, Davrondzhon & Helkala, Kirsi & Soendrol, T. (2006). Gait recognition using acceleration from MEMS. 2006. 6 pp.-. 10.1109/ARES.2006.68.

7. Mjaaland, B. B., Bours, P., & Gligoroski, D. (2011). Walk the Walk: Attacking Gait Biometrics by Imitation. LECTURE NOTES IN COMPUTER SCIENCE, (6531), 361.

8. Casale, P. Pujol, O. and Radeva, P. 'Personalization and user verification in wearable systems using biometric walking patterns' Personal and Ubiquitous Computing, 16(5), 563-580, 2012

User Device Interaction Verification Survey (UDIVS)

Abstract— **Within the last decade bio-metric authorization and verification systems have become increasingly popular on mobile devices. Verification systems such as fingerprint scanning, face-detection, and in some cases retinal and iris scanners are becoming more feasible on everyday computing devices. However, these systems are still not as popular as memory based pin passwords, and are prone to repeated spoof attacks. If a wolf is capable of verifying themselves once with these methods there is a good chance the wolf can get into the system again. The only exception to this is the memory based passwords which can be rest. But these pins and passwords can easily be exhausted out by most dictionaries. Due to these shortcomings there is a need for a system that does not easily allow re-entry to a wolf if they crack the system, while at the same time be comfortable enough by everyday user's to be adopted and preferred over the traditional based pin and passwords. Our solution to this is a User Device Interaction Survey (UDIVS) that asks questions to the user about their recent activities to verify them as a genuine user of the system, or deny them entry if they are an imposter. Extracting data from the user and asking security questions about daily activities maybe a more effective way to abate spoofing attacks, while also being user-friendly and low costing. Utilizing data that we collected from user device interactions and developing a score level fusion we found. From our research we have found that such a system is feasible. We were able to develop a system that offers a d-prime value of 0.97 and a Equal Error Rate of 0.3. Although, these numbers are not astonishing there are many reasons to believe that these numbers can be improved immensely if provisions are met, and at worst case the system can be utilized in conjunction with another modality such as key stroke and gate analysis which at the moment does not have a useful and popularized implementation.**

I. Background

Separate studies have been conducted to explore different methods that strengthen knowledge-based logins on mobile devices. They also discuss areas of multimodal systems that are commonly researched and developed.

Grant Ho explores more features than previously implemented in the case of mobile devices and is the first one to apply them to short 4-digit pin codes[1]. Key features include: the duration of each key tap, the latency of each key tap, the size of each key tap, and accelerometer readings over the course of logins. Hos reasons that the accelerometer could give insight into the phones orientation in the hands of legitimate users during logins. His experiment yielded remarkable results with a FAR of 5.6% and an FRR of 7.6%. Even if an imposter guesses the PIN correctly, their keystrokes would ultimately trigger a rejection by the system. Although the paper considered attackers who are able to guess easy, typical PINs, it did not discuss problems with attackers who have stolen the PINs and given the opportunity to mimic keystroke behaviors.

Zhen et al tested the feasibility of tapping behaviors on password verification[2]. The researchers extracted acceleration, pressure, size, and time data from tap inputs. Using taping dynamics in conjunction with pass-codes resulted in accuracies with equal error rates down to 3.65%. However, user behavior changes, such as a broken thumb, could render their system infeasible requiring a reset of the system. Pass-code changes due to security reason will ultimately reset the dataset for machine learning algorithm.

Yu et al proposed an interesting alternative to traditional password[3]. The researchers developed a mobile 3D authentication system where users draw lines to cubes using a leap motion device in virtual space. Each cube is presented in virtual space. The sequence of lines drawn is recorded and fed into an algorithm that produces a unique password. In its current state, this method appears awkward and impractical in a normal setting as it requires a separate apparatus in a controlled environment.

However, users are not familiar with these techniques, and it is unclear whether general users will become habituated with these modalities. This is why we came up with our system. We believe that it is safe to assume that most users will feel comfortable answering multiple choice questions. Also, (UDIVS) utilizes a sliding window approach. This allows old data to be removed from the system as new data comes in, which means that the space complexity of the system as time increases remains constant, this means that it is entirely feasible to keep all the data on the device locally without compromising a user's privacy. It is also worth mentioning that a perfect system does not exist. Any software or hardware security system is capable of being circumvented.

II. INTRODUCTION

One of the largest concerns in bio-metric verification is generating a high performance, reliable, and low cost verification system, which we hope to satisfy with (UDIVS). Knowledge-based recognition is still widely used for typical logins, but it is ultimately plagued by shoulder surfing, smudge attacks, and other spoof methods. There is a need for a system that is both adaptable and flexible for the everyday portable device users that reduces the risk of using the same verification method over and over again. UDIVS utilizes a score level fusion for matching genuine users to their device/s, if the score satisfies a predefined threshold the user can be verified as a genuine. The final score is generated from the scores accredited from a set of questions. A question may or may not have multiple answers and will change after each login attempt. Questions could be asked

about the sequence of geo-locations a user may have visited. Weather or not they walked or drove to a geo-location. which apps they used and when. What networks they were logged into, what blue tooth devices they used recently ect.

III. Introduction to System Development and Measuring Question Effectiveness

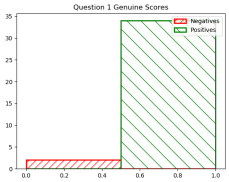
A. Question Generation

In our system we collected data from the day to day device interactions of two users. These users were able to generate over 4,000 combined data samples. These samples tracked Geo-location and phone activities. From this data we generated six different questions:

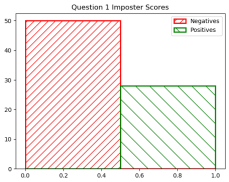
- Question 1: "Which app did you use most recently?"
- Question 2: "What place were you at most recently?"
- Question 3: "Which place were you at around**[insert time]**"
- Question 4: "Which of these places did you go to yesterday?"
- Question 5: "How long were you on this app **[insert app]**?"
- Question 6: "Which of these apps did you use most frequently so far today?"

The user then must select the correct answer(s) from a multiple choice list. The option list must not have a repeated correct answers. From these questions, we asked the genuine and imposter users 3 questions per a log-in attempt and collected data on the amount, of the three questions, they were able to answer correctly. The questions were not repeated and were drawn randomly from the list. We collected data on 73 genuine user attempts, and 146 imposter attempts. It is important to note that there is no criteria for question generation as long as there is enough data to generate correct and incorrect answers unambiguously, and the answers should not be easily guessed. For example, our system does not utilize Wi-Fi or Blue-tooth device activity; one could implement such a system as long as correct answers and incorrect answers exist. But problems could arise if users only utilize one Wi-Fi network or one Blue-tooth device. In theory you could ask short answer questions, but we decided not to do this because users want quick access to the system and if the imposter is able to answer this question correctly they could answer it correctly over and over again because the correct answer does not change with time, making the answer predictable. This, ultimately defeats the purpose of such a system. The goal is to not allow repeated unauthorized entry. Some of the questions we have implemented have performance issues do to predictability and we can demonstrate this issue in the following figures. The red shows the total amount the genuine or imposter incorrectly answered the question, and green shows the total amount of times the genuine or imposter was able to answer the question correctly.

Despite these graphs being intuitive they don't capture the performance of a question relative to each other. For instance, since these graphs look at totals rather than percents they

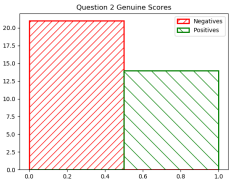


(a) Genuine

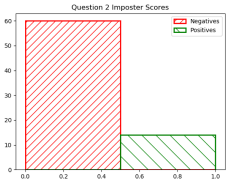


(b) Imposter

Fig. 1: "Which app did you use most recently?"

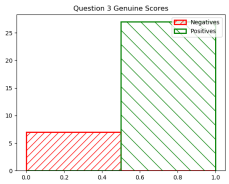


(a) Genuine



(b) Imposter

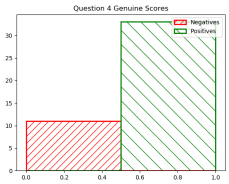
Fig. 2: "What place were you at most recently?"



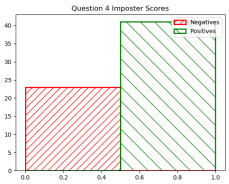
(a) Genuine

(b) Imposter

Fig. 3: "Which place were you at around **[insert time]**?"

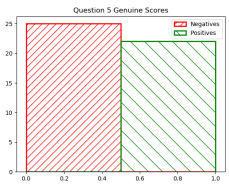


(a) Genuine

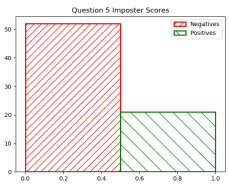


(b) Imposter

Fig. 4: "Which of these places did you go to yesterday?"

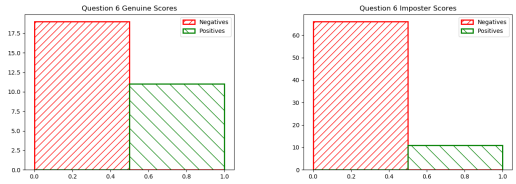


(a) Genuine



(b) Imposter

Fig. 5: "About how long did you use **[insert app]** for?"



(a) Genuine (b) Imposter

Fig. 6: "Which of these apps did you use most frequently so far today?"

can be misleading since there is a disproportional amount of genuine and imposter users. By considering incorrect and correct answers as binaries, we developed a simple way to describe question accuracy in following equation:

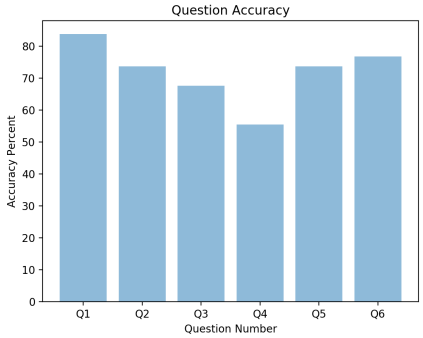
Question Accuracy =

$$((G_{correct} + I_{incorrect}) - (G_{incorrect} + I_{correct}))/Q_{Amount}$$

contents

Where G is the genuine user, I is the imposter, and Q is the question. All relative to the specific question that accuracy is being calculated for. A chart that describes the performance of all six questions we collected data on can be seen in Figure 7.

Fig. 7: Chart of Question Accuracy



With this we are able to see that Question 4 performed poorly compared to its contemporaries. This makes sense when looking at Figure 4. It is easy to see that The imposter was able to answer this question just as often at the genuine user. Which leads us to another important distinction. Despite the Question Accuracy being low, this question is not necessarily a bad question. Through the many trials of developing questions we came to understand that these performance metrics are better utilized to try to understand why a question is performing poorly and what can be done to improve its performance rather than not asking a question at all. These metrics are useful to see if progress is being made. Many factors must be considered when developing a question and understanding why a question is under performing such as:

- The correct answer are predictable
- The user for this particular device is habituated.
- Data mining on the users device activities and geo-location.
- Ambiguity of the question.

1) The correct Answers are Predictable: In our case, often times the correct answer of "Home" would appear for Question 4. This is an easy target for someone trying to break into the system. This is a result of a bug in our program and our thought process of trying to develop questions. In a multiple choice system, we elected to not have more than one correct answer due to the fact that adding this feature would be too time consuming for the nature of this project. Since a question needs to have multiple incorrect answers, there may not be enough data on the user to ask this question properly. In the case of Question 4 "Which of these places did you go to yesterday?", if one of the options is home, then it is obvious to both the genuine and imposter that home is the correct answer. But the system generated this question in the correct way. Maybe the user never left home yesterday, or maybe the user visited five different locations with home being one of them and there was not enough data to generate the incorrect answers on the other locations, since we can't have more than one correct answer, it was more logical for the system to generate home as a correct answer than making it an incorrect answer for another location.

2) The user for this particular device is habituated: In this case it is not a matter of the question being a bad question. The question is bad for this particular user which is an important distinction. In the case of question 4. The user may visit the same places everyday. This allows a imposter to easily guess the question correctly if they personally know the person. In this case, it is better for the system to not ask this question until the habit breaks.

3) Data Mining User: In the digital world we live in today, it can be easy to break a system such as this with data mining tactics. Since a sliding window approach is used it would not be costly at all for a data mining predator to spoof the system. Some of the issues with this also has to do with the user utilizing a significant amount of social media apps where it is easy for attackers to collect data on geo-location and device activities that link to a social media account.

4) Ambiguity Of The Question: Sometimes the question is poorly phrased and confuses the user. In the case of Question 2. "What place were you at most recently?", we should have specified the place before your current location. This question embarrassingly confused us despite being the developers of the system, which ultimately hindered it's performance.

B. Improving Questions

Although we did not get a chance to implement these solutions, it is worth mentioning some strategies that could be utilized. Some of the ideas that we thought of are:

– Meta Questions: Collecting data about previously asked questions and asking the user about them to verify them, which is something that cannot be data mined.
– Machine Learning: To ask *"smart questions"* so that habitual users are not compromised
– Multi-Modal: *"Simon Says"*

In the next section we will analyze the over all performance of the system as a whole and how the above ideas can boost performance of individual questions as well as the overall system.

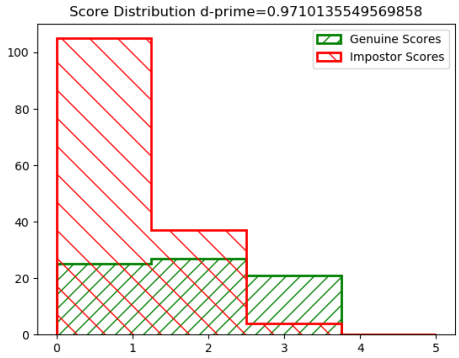
IV. Results and Analysis

A Detection Error Tradeoff (DET), Receiver Operating Characteristic (ROC), and score distribution is useful to analyze the performance of the UDIVs. In our evaluation, there are two underlying values to describe the accuracy of genuine and imposter detection.

- D-prime (D): quantifies the separation between genuine and imposter score distributions. A higher number is better.
- Equal Error Rate (EER): the point at which the False Accept Rate (FAR) and False Reject Rate (FRR) are equal. (Lower is better)

In the following, we present the efficacy on a naive hybrid verification system, the highlighted performance from our pool of questions, and ways to improve the performance of UDIVs.

Fig. 8: Score Distribution Curve with D'

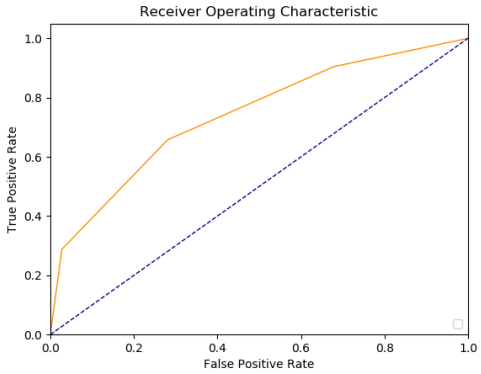


A. Verification Accuracy

Figure 9 and 10 both have an EER of 0.31. which means the overall UDIVs is more likely to reject an imposter than a genuine user with a roughly 60% accuracy. The score distribution x-axis is actually index from 0 to 3 to represent the total questions answered correctly from each survey attempt; anything outside of this range is white spaced for visual clarity. Observing the score distribution of the imposter and genuine attempts, we notice that imposters are very likely to answer less than 2 questions correctly while genuine users tend to answer 2 or more questions accurately.

Fig. 9: Detection Error Tradeoff (DET)with EER

Fig. 10: Receiver Operating Characteristic (ROC)



The D-prime is .97(Fig. 8) suggesting there is a slightly low distinction between imposter and genuine attempts.

Based on these scores, we can make several conclusions. Imposters are almost unable to answer 3 questions correctly, but typically answered below the threshold 60% of the time while Genuine users passed the threshold 60% of the time. We can confidently say that Genuine users have a good prospect of answering questions 1, 3, and 4 correctly. Figures 1 and 6 show that Questions 1 and 6 are particularly resilient against attackers. Additionally, observations from figure 7 acknowledges that question 1 and 6 have a high accuracy relative to the genuine user; question 1 achieves the highest accuracy above 80%. This suggests that questions 1 and 6 excels in performance with regards to safeguarding spoof attacks but are intuitive enough for genuine users to answer accurately.

With the current threshold at 2 questions, the FAR and FRR are somewhat higher than expected. Stunningly, we can observe a significant drop in the imposters frequency of answering 3 questions correctly. In light of this, having a threshold greater than 2 is desirable to dramatically reduce the FP, however this inversely increases the FN. Users will have difficulties answering more than 2 questions correctly. This is actually a positive outcome because we can expect a critical decrease in FP by increasing the threshold to one more question. As for the genuine user performance,

there are several areas we can make adjustments to that enhances the likelihood of correct verification discussed in the following paragraphs.

B. Independent Variables

We have identified 2 feasible independent variables, one of which was intentionally determined; the other stems from analyzing the results of our experiment. For the UDIVS to remain updated and robust against data miners, we implemented a sliding window approach of a single day of raw data. This method functions as a feature selecting the most relevant data to answer the question. Naturally, genuine users are more likely to recall recent data. Because of this we expected good performance from the genuine users end. All but question 4 performed well. This is the only question that relies on the users location data from yesterday, and coincidentally has the lowest genuine user accuracy. All other questions employ a sliding window approach of the current days data. It is possible here that the sliding window approach may have contributed to the unexpected lower performance of question 4.

Our second independent variable is increasing the number of questions. Previously we discussed that there is a prominent visual decrease in successful imposter attempts from 2 to 3 answers from figure 8. What this entails is that increasing the threshold value from 2 to 3 will dramatically reduce the FP. As of now, our system only asks 3 question. We can increase the number of questions asked so that the threshold can be increased accordingly. We theorize that changing the number of questions can positively influence the performance of UDIVS, but further testing is necessary to make any conclusions. If this concludes to be true, we can manipulate the number of questions to query based on the security level of the mobile device. A higher question count will be associated with heightened defenses against if there is suspicion of an attacker, or unnatural behavior interactions with the device have been detected.

C. Issues and Methods to Improve Performance

Certain aspects of our system can be changed to theoretically improve FARs and FRRs. For example, question 5 proved to be challenging for genuine users to answer, but the reason is because there is a single correct answer for each question. Questions 5 asks How long were you on this app?. If the user provides an answer of 0-10 minutes when the actual answer is 11 minutes, then the correct answer is 10-20 minutes. The user gave a reasonably honest estimate that is relatively close to the actual answer. Instead they are penalized. Question 6 asks Which of these apps did you use most frequently today?. The answer is generated based on the frequency of opening an app when it should also account for the duration of use. The questions themselves must be tweaked to accommodate human-like estimates and rationale to improve correctness. To remedy this, we can implement a weighted score averaging where each answer is given a weight based on relevancy. A higher weight is associated with the most accurate answer. Find the average score and test it against the threshold (which can take the form of decimal/fraction numbers).

The dataset used came from a third-party android application, SmarterTime[5]. This means we do not have full control of the data collection process. For example, the dataset contains a duration of each application used. The duration only pertains to the length of on-screen activity. If the genuine user were to use a music media player, such as Spotify, then normally the user would turn off the screen of their device when listening to music or an eBook. If the user is queried with question 5, they would assume the length of music played on Spotify is 40 minutes, but the actual data recorded only 7 minutes for the time it took the user to search for a playlist. Having uncontrolled data collection causes some ambiguity in the answering process of certain questions, but it is not necessarily a disadvantage. The specificity of answers can be utilized to generate more contextualized questions. For example, we can add a question such as What was the last app you opened at [insert location] instead of What was the last app you used. The user will clearly understand the questions premise is the last opened application.

Different Imposters have different probabilities of bypassing UDIVS. Without having 100% controlled imposter subjects, our results are impartially influenced. One subject is sure to know a decent trend of the genuine subjects whereabouts. The genuine subjects are also uncontrolled. We observed a noticeable increase in the FAR when the subjects remained sedentary in the same location everyday within the time-span of the experiment. Figure 3 and Figure 4 primarily deals with location data, and here we see imposters with the highest False Positive scores (green) out of all 6 questions attempted. The effectiveness in deterring imposters is highly dependent on the genuine users variability and unpredictable habits along. The relationship between the genuine user and imposter subjects can influence the effectiveness of UDIVS. This factor cannot easily be controlled. Instead, we can categorize the effectiveness based on a Threat Level Model that classifies four levels of knowledge the intruder is expected to possess about the genuine user.

D. Meta Questions

With a dynamic question system, it will be difficult for attackers to break into the UDIVS system unless they have some prior knowledge of the genuine user. We must consider the possibility that some attackers rely on data mining to collect a users data on certain popular apps such as Facebook and Twitter. These attackers pose a great threat if they can utilize stolen data to effectively guess the correctly. To contest their efforts, we can introduce a concept of Meta Questions that adds a layer of complications to the attacker. For instance, an attacker can data mine the users geo-location, and other activities linked to social media. When attempting to infiltrate the UDIVS, they are prompted with a question that asks, What was the last question you answered?, therefore the attackers mined data is rendered useless on this type of question. Theoretically, the amount

of entropy has theoretically increased since the data miner would need access to the UDIVS program, further making this system more exhausting to spoof.

E. Machine Learning

Sometimes, UDIVS may generate questions that are unintuitive. For example, if the user develops a repetitive habit of going to the gym then returning home, UDIVS will likely generate home or gym as the answer to location-based questions. These answers are too obvious since everyone returns home at some point, and many people workout consistently. In our case, home and C4 Lab were the most common answer choices since the subjects spent most of their time in these locations. Other cases like insufficient data can interfere with quality questions. Some users may not have experienced enough variability when they begin using UDIVS. Machine Learning can come into play to detect patterns of habit and omit weak answers from questions, or even dismiss the question entirely. It may also choose smart questions (questions with enough variance in the dataset) to query the user based on the amount of data available, similar to feature selection.

F. Multi-Modal

Integrating multiple biometric modalities can potentially strengthen UDIVS verification accuracy. As demonstrated by Grant Ho[1] and Zheng et al[2], keystroke elements like duration, latency, size, and pressure can assist in distinguishing genuine users from imposters. Other modalities can be considered; fingerprint, iris, gait, facial, etc., where users can be prompted in a Simon Says fashion. The UDIVS can prompt the user for a random biometric trait to present as part of the question set. In doing so, questions should be less predictable to the queried user, and possibly impossible for the average imposters. For an attacker to bypass this system, they must obtain cross-biometric samples of the genuine user, thereby dramatically increasing the entropy of spoofing. Keep in mind the three methods previously mentioned are theoretical insight into the potential expansion of UDIVS that is expected to improve its overall performance.

V. Discussions

Incorporating UDIVS system for mobile verification is user friendly, high performance yielding, reliable and cost effective. Because our system is dynamic, it keeps generating questions based on the recent activities, negating the susceptibility of a long used passcodes/pins. The performance remains constant throughout the lifetime of the system. There is no way the system is ever out of date because of generality. Improvements can come in ways of coming up with intuitive ways to survey the genuine user. This is different from any other methods of verification and can include other modalities. Thus resulting in a Multi-modal system that is robust and ultimately secured.

A. Advantages of UDIVS

Grant hos method doesnt factor in possibilities that behaviors can be picked up. Though it includes more features than ever to ensure that the memorization of pass-code is not the only obstacle, it just added another layer of protection through keystrokes which can also eventually be broken into. UDIVS is resilient against mimicry because of its dynamic nature which is not something an imposter can learn.

Zhen et als method for tapping dynamics along with passcodes improved results, but the system comes to a halt when theres a situation which disables the users way to tap. For any inconvenience when the user cant provide necessary input, like in case of a broken thumb, the system needs to be reset. If machine learning is concerned, that would wipe the whole progress away. While in our case, the surveys are assembled using a sliding window approach. This eliminates the need for storing data for longer periods of time and resetting the system.

Ye et al devised a method which is innovative but that puts users in awkward positions which is not something everyone can enjoy. The method for answering questions in UDIVS is familiar and user-friendlier.

VI. Conclusions

A. What We Accomplished and Learned

The accomplishment of our project is that we presented a hybrid verification system which is feasible and convenient. We faced several scenarios during the project. First being, the system makes it difficult for genuine users to grant imposters permission for accessing the device(s). The process is rigorous and requires the genuine user to be present in some shape, way or form during the verification. Moreover, the questions context should reflect what the dataset collects. For example, an app can be running even while the phone is locked, so the recently used app may not precisely be the last opened one. Similarly, it can vary depending on the locked and unlocked states of the phone. We also learned that the questions could be misleading to answer if the users are not aware of the logic behind it.

B. What To Do Differently

If the time span was a bit stretched there were scopes where we believe we could improve upon. First of all, we would have liked our data to be modeled per our need. In order to do that, we could develop an android app just for data acquisition instead of relying on third party app. Our questions are now pretty straightforward. Then again, if we had more insightful data, we did not have to tailor our questions and could come up with more thorough ones with multiple right answers. This is one other area wed like to do differently. Due to time restraints, our implementations focused on getting the system running with one right answer per question. But in cases, where there are multiple right ones, weights could be given to the options to increase the relevancy of certain ones, and the scoring fusion could reflect that. For example, for the question which of these places did you visit yesterday, we could include more than one places

from yesterday. The weights could indicate the priority of the obvious ones vs the less obvious ones and make the scoring continuous.

C. Limitations

There are few areas our system can still fall behind. But it is important to note that, these are not downright flaws, these are just rooms where this new concept can grow more into eventually. If a user attempts multiple log-in attempts in a short span of time, the same questions can keep appearing. This will help the imposter break into the system if hes been in close contact with the genuine user. So the time interval between each attempts could cause risk of repetition. There should be enough activity during each session to provide a rich source of data for later. At this stage of our project, the questions are static. They are being chosen from the same set of questions for every user. There is also the discrepancy in user habits. If the genuine user experience very little variation in their daily activities, then imposters can spoof the systems with more ease. In the future, machine learning could tackle the job of handling users based on their own set of behaviors and devise challenging situational questions.

ACKNOWLEDGMENT

University of South Florida Computer Science and Engineering Department and Dr. Tempestt Neal Biometric Authentication on Mobile Devices/Biometrics Instructor

REFERENCES

[1] G. Ho, TapDynamics: Strengthening User Authentication on Mobile Phones with Keystroke Dynamics. In Stanford University, 2013.

[2] N. Zheng, K. Bai, H. Huang, H. Wang, You Are How You Touch: User Verification on Smartphones via Tapping Behaviors In IEEE Int. Conf. on Network Protocol, Dec. 2014.

[3] Z. Yu, I. Olade, H. N. Liang, C. Fleming, Usable authentication mechanisms for mobile devices: An exploration of 3D graphical passwords, In Proc. Int. Conf. Platform Technology and Service (PlatCon), 2016.

[4] T. J. Neal and D. L. Woodard, "Spoofing analysis of mobile device data as behavioral biometric modalities," 2017 IEEE International Joint Conference on Biometrics (IJCB), Denver, CO, 2017.

[5] SmarterTime [Mobile Application Software]

Final Project Report

Introduction

The objective of the project was to implement a biometric authentication system using keystroke inputs. As keystroke inputs are easily acquired from a user, and are continuously acquired in most cases, they would be well suited for continuous biometric authentication systems. Notable features of the system designed are feature extraction using variance threshold of input data, and score level fusion of two machine learning matchers: k nearest neighbors and naive bayes classification. Also, The number of training samples was altered to determine its effectiveness on the systems results.

Background

Analysis of keystroke based systems has been done extensively in the past. In the paper *"Keystroke dynamics authentication for mobile phones,"* the authors collected keystroke data during password entry, and concluded that "if a strong secure authentication scheme for mobile devices is needed, it cannot rely exclusively on keystroke dynamics"(Maiorana et al. 25). Another paper, "Performance of a Long-Text-Input Keystroke Biometric Authentication System Using an Improved k-Nearest-Neighbor Classification Method", noted that using k nearest neighbors algorithm offered advantages in terms of flexibility compared to other methods. The paper also suggested there could be further work attempting to improve the EER of keystroke biometric systems, particularly with regards to database size(Zack et al.). With this in mind, the project implements two matchers: k nearest neighbor and a naive bayes in an attempt to overcome some of the weakness mentioned by the first paper, with k nearest neighbors being usd because of its recommendation by the second. Bayes was chosen due to prior work involving it. It was decided to vary the training set size to investigate the proposed avenue of further research mentioned by the second paper.

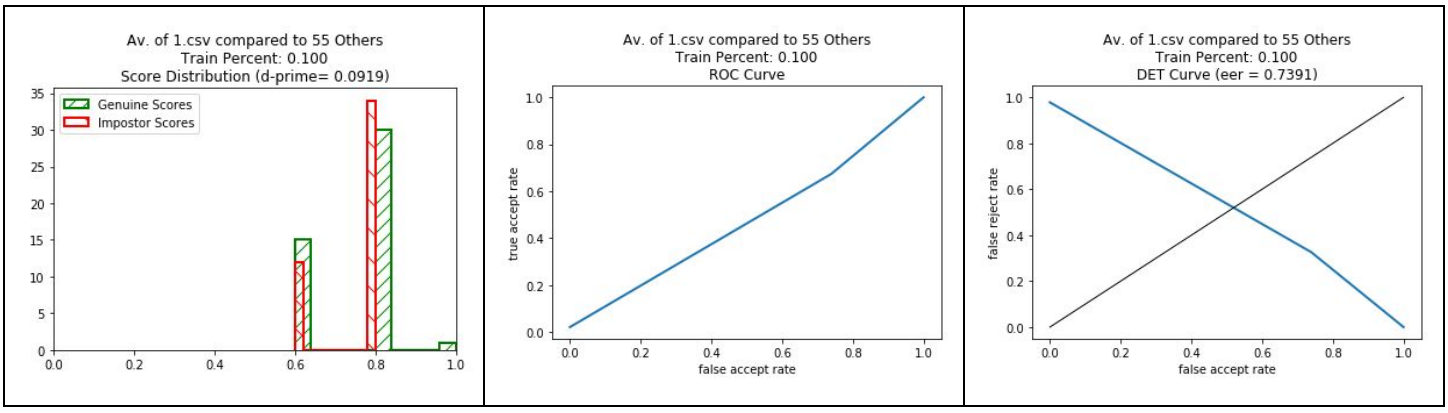
Method

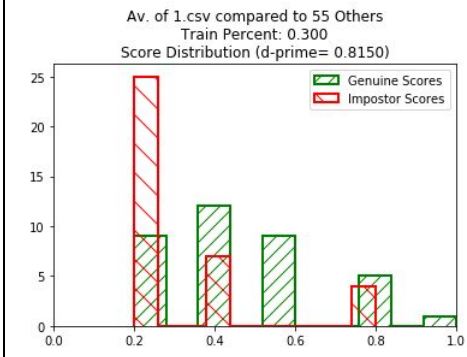
Our biometric system uses the keystroke modality and first acquires data from two persons: a genuine user and an impostor. Feature selection was then performed on these sets using a Variance threshold comparison method, in which any feature column with low variance was thrown out. This is advantageous because higher variance yields more information when used as a training sample than a feature with low variance in machine learning algorithms. The columns pruned from the genuine users dataset are also pruned from the impostors.

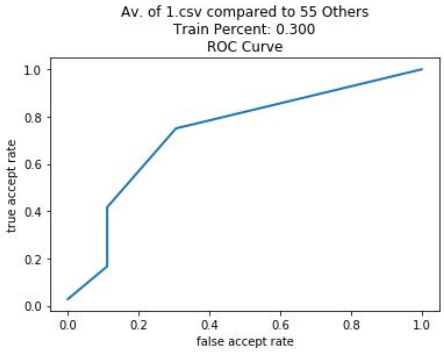
For use in the machine learning algorithms, the data is then divided up into training sets and test sets for both the imposter and genuine user data, with the specific amount to be assigned to the training set being varied per run to see how results are affected. The range of values used for the training set percentage are: 10%, 30%, 50%, 70%, and 90%. These training sets are then combined and used to calibrate the two machine learning matchers used: k nearest neighbor and naive bayes. K nearest neighbor matches an input to the closest training set, while bayes uses probability to match an input to an output. Both matchers are trained using the training sets, then tested using the test sets to derive results from a genuine user and an impostor user. As multiple matchers are used score-level fusion is employed to combine the results, in this case the average of the two outputs is used to get an overall result. The overall result is then used to construct a ROC curve, DET curve, EER, a d-prime value, and score distribution graphs. A single genuine user is selected and compared to all other users provided in the dataset. This means that a single user is compared to 55 other samples. The results are averaged together then the ROC curve, DET curve, and score distribution charts are produced seen in Graphs A and Graphs B in the Results section.

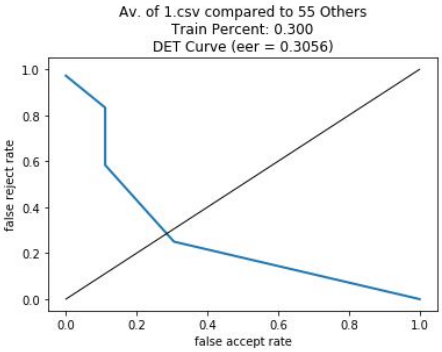
Results

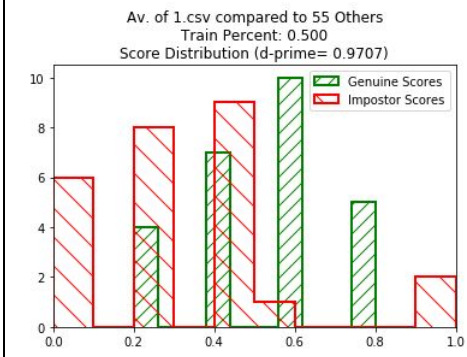
Graphs A: Comparing 1.csv all other samples then averaging the results:

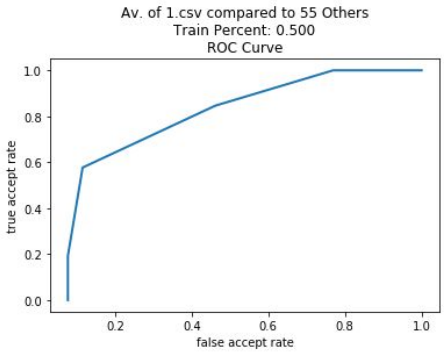


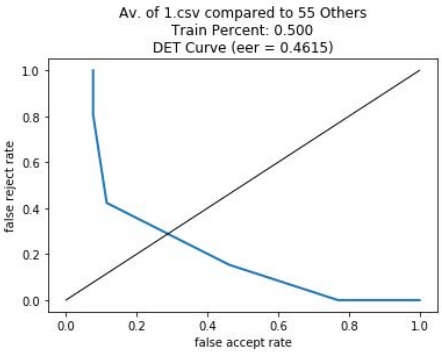

Av. of 1.csv compared to 55 Others
Train Percent: 0.300
Score Distribution (d-prime= 0.8150)
Genuine Scores
Impostor Scores

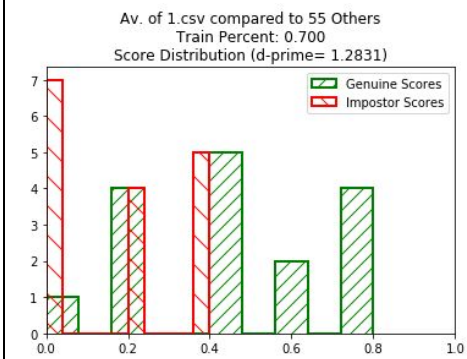

Av. of 1.csv compared to 55 Others
Train Percent: 0.300
ROC Curve
true accept rate
false accept rate

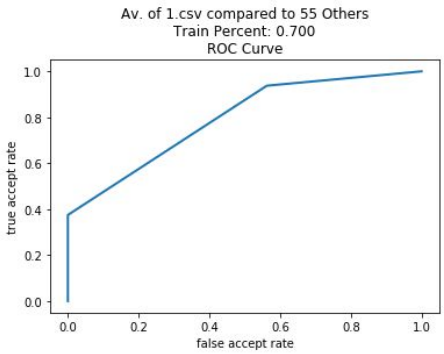

Av. of 1.csv compared to 55 Others
Train Percent: 0.300
DET Curve (eer = 0.3056)
false reject rate
false accept rate

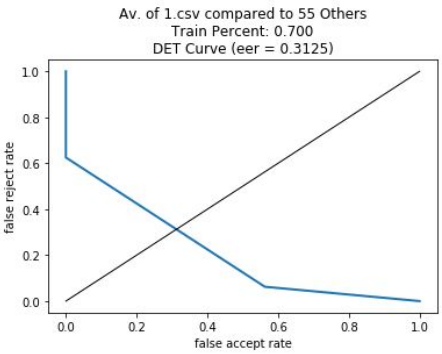

Av. of 1.csv compared to 55 Others
Train Percent: 0.500
Score Distribution (d-prime= 0.9707)
Genuine Scores
Impostor Scores

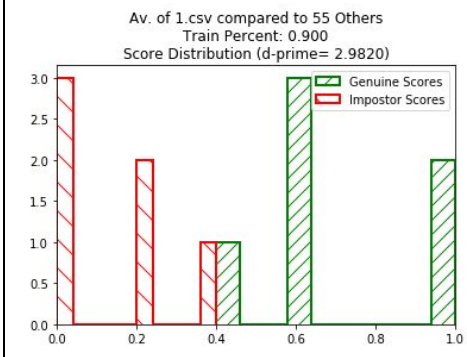

Av. of 1.csv compared to 55 Others
Train Percent: 0.500
ROC Curve
true accept rate
false accept rate

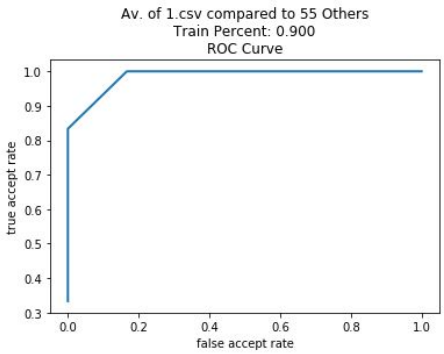

Av. of 1.csv compared to 55 Others
Train Percent: 0.500
DET Curve (eer = 0.4615)
false reject rate
false accept rate

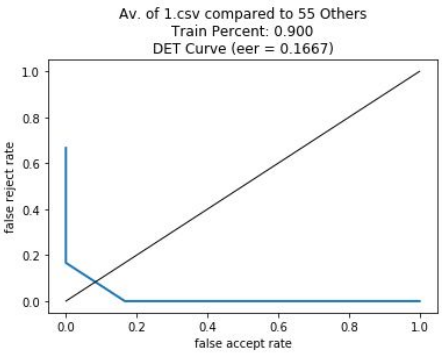

Av. of 1.csv compared to 55 Others
Train Percent: 0.700
Score Distribution (d-prime= 1.2831)
Genuine Scores
Impostor Scores


Av. of 1.csv compared to 55 Others
Train Percent: 0.700
ROC Curve
true accept rate
false accept rate

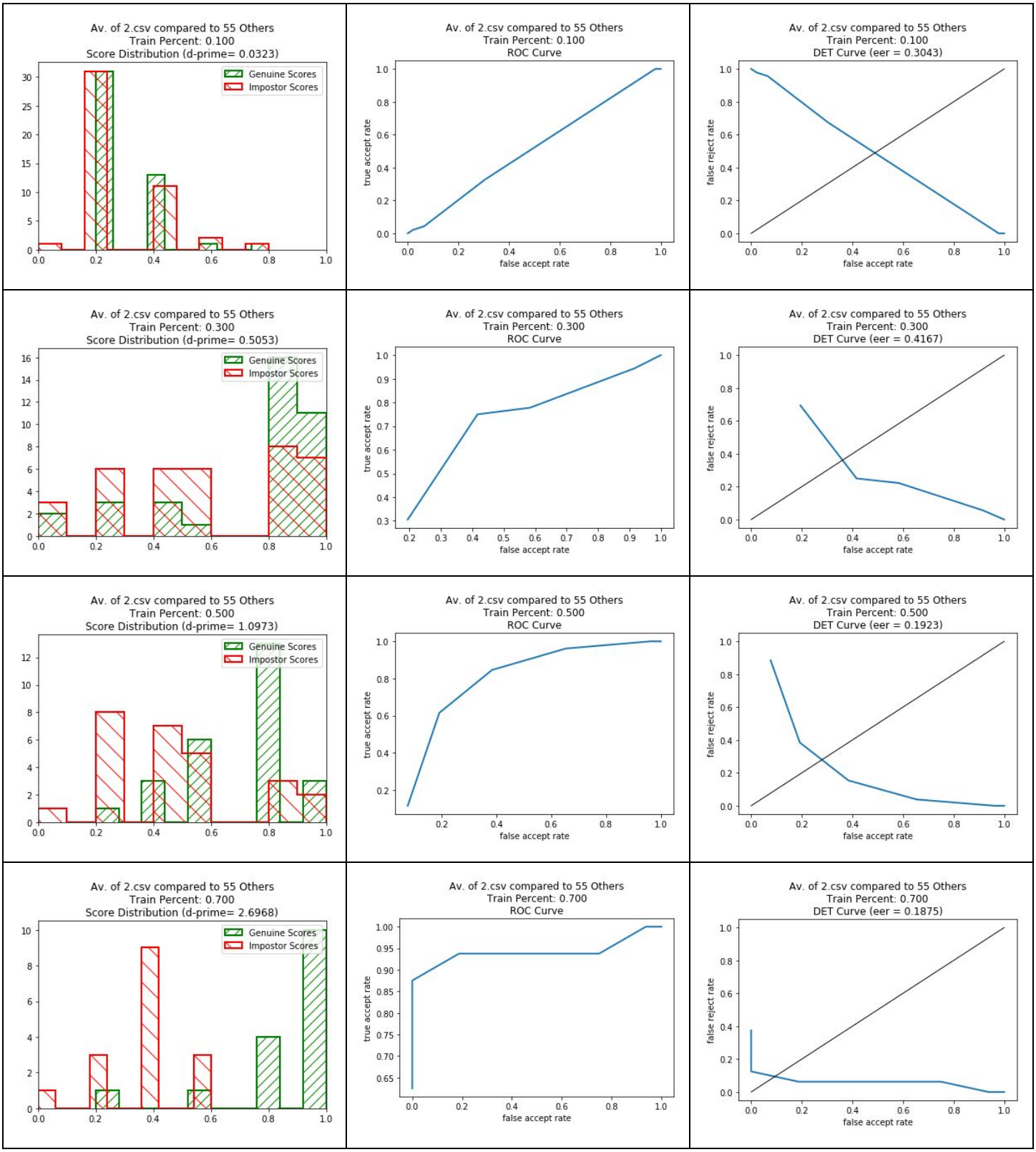

Av. of 1.csv compared to 55 Others
Train Percent: 0.700
DET Curve (eer = 0.3125)
false reject rate
false accept rate

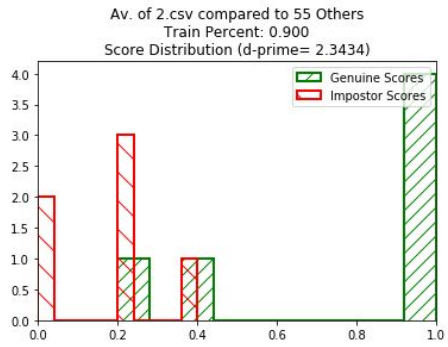

Av. of 1.csv compared to 55 Others
Train Percent: 0.900
Score Distribution (d-prime= 2.9820)
Genuine Scores
Impostor Scores

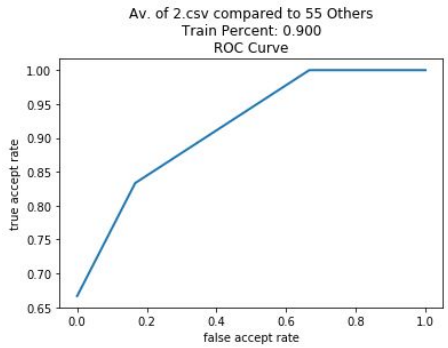

Av. of 1.csv compared to 55 Others
Train Percent: 0.900
ROC Curve
true accept rate
false accept rate


Av. of 1.csv compared to 55 Others
Train Percent: 0.900
DET Curve (eer = 0.1667)
false reject rate
false accept rate

Graphs B: Comparing 2.csv all other samples then averaging the results:




Av. of 2.csv compared to 55 Others
Train Percent: 0.900
Score Distribution (d-prime= 2.3434)
Genuine Scores
Impostor Scores


Av. of 2.csv compared to 55 Others
Train Percent: 0.900
ROC Curve
true accept rate
false accept rate

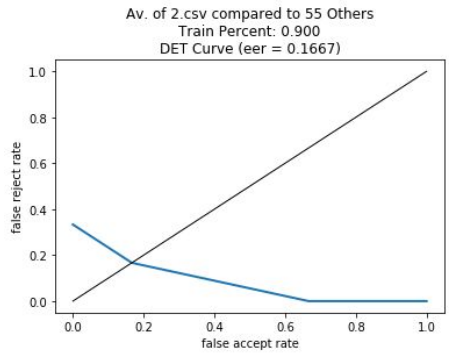

Av. of 2.csv compared to 55 Others
Train Percent: 0.900
DET Curve (eer = 0.1667)
false reject rate
false accept rate

Figure A: 1.csv vs 55 others: d-prime and eer for 5 values of Train Percent

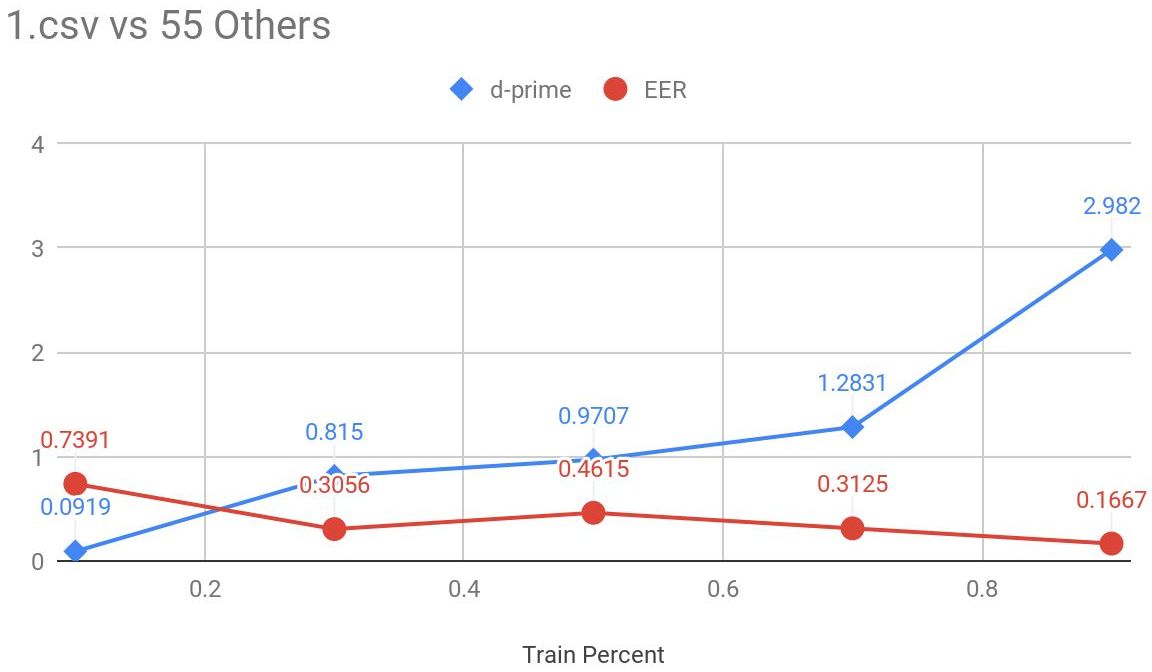
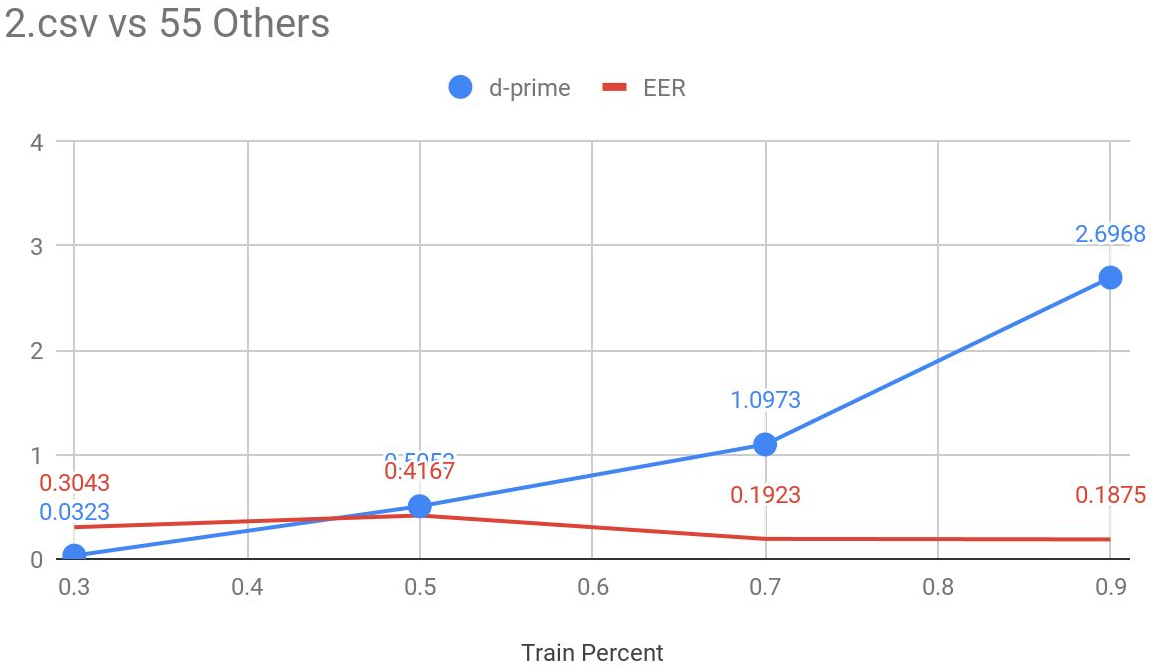


Figure B: 2.csv vs 55 others: d-prime and eer for 5 values of Train Percent



Discussion

The most obvious conclusion from the above charts is that as the training set percent increases, the EER decreases. Additionally, as the training sample percent increases so does the d-prime value. Both of these results suggest that as the model has more samples to train with, the more accurate it becomes.

Summary

The objective of the project was to implement a biometric authentication system using keystroke inputs. This was accomplished successfully. Incorporating multiple matchers seemed to make the system more secure, and as more training samples were used the more accurate the systems output became. Clearly our biometric system returned positive results, especially when using a higher number of training samples. However, we must consider the results from our background section in which the paper *"Keystroke dynamics authentication for mobile phones,"* the authors concluded that "if a strong secure authentication scheme for mobile devices is needed, it cannot rely exclusively on keystroke dynamics." This means that despite the positive results of our biometric system we must consider the ease of which a keystroke authentication system can be attacked. This is outlined in the the paper *"Snoop-Forge-Replay Attacks on Continuous Verification With Keystrokes,"* in which the authors highlight the ease of which keystrokes can be collected and replayed to trick a biometric

system. The most concerning aspect is that the authors found that keystrokes can be collected and used at a future date.

One way to strengthen our keystroke authentication system against replay attacks would be to implement a method where the words are analyzed for common spelling mistakes. If the biometric system could consider a user's misspelling patterns as a measure of authentication this could prevent some replay attacks. Additionally, specific words and phrases could be used to better identify the genuine user. Alternatively, a multimodal system could be implemented where keystroke and user device information are considered together to further strengthen the system.

References

Maiorana, Emanuele, Patrizio Campisi, Noelia González-Carballo, and Alessandro Neri. "Keystroke dynamics authentication for mobile phones." In Proceedings of the 2011 ACM Symposium on Applied Computing , pp. 21-26. ACM, 2011.

Zack, Robert S, et al. "Performance of a Long-Text-Input Keystroke Biometric Authentication System Using an Improved k-Nearest-Neighbor Classification Method." 2010 Fourth IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS) , 2010. IEEE Xplore , doi:10.1109/BTAS.2010.5634492.

Snoop-Forge-Replay Attacks on Continuous Verification With Keystrokes. IEEE Transactions on Information Forensics and Security. Rahman, K. A., Balagani, K. S., & Phoha, V. V. (2013).

Gait Authentication

Introduction

Biometric authentication is used in many cool applications such as security systems and for verification in many more systems. Some of the cool applications that are used is mainly for identification of people using gait. Gait can also be used authenticate continuously on a mobile device versus only at the beginning. Data is collected using sensors especially wearable sensors for recording gait.

We will compare and see how smoothed and unsmoothed data affect actual gait recognition and matching. In this paper we are going to mainly discuss about gait used as a biometric modality and the technique used in matching would be Distance Time Warping (DTW). The various steps that will be used in gait recognition are data collection, noise reduction, feature extraction and matching.

In our project the data is already given to us,we are using different noise reduction techniques after which the data is sent for histogram and features extraction and matching.

Background

The paper that motivated our project was the article, “Walk the Walk: Attacking Gait Biometrics by Imitation”. This article mentions the study of imposters mimicking human gait to attack the biometric system. While attackers were able to attempt to mimic the victim’s gait based on the video, the harder they tried the more they would fail and plateau at a certain point. Although a couple were able to get close to the victim’s gait without trying.

Even when gait mimicking of the imposter is similar to the genuine subject’s gait, it is important to distinguish the differences between the imposter and genuine subject in order to identify the correct individual for authentication purposes.

Method

The approach to the problem will involve trying to find the closest matches to the actual genuine subject to see whether they could later be imitated and whether we need to improve our security.

The design of the experiment first started taking in the raw gait data. We will be using the raw gait data provided from the gait_data folder on Canvas. This data was obtained from an Android smartphone positioned in the chest pocket of the participants. There are 22 participants and 22 excel files corresponding to that participant with

accelerometer data:

---time-step,x acceleration,y acceleration,z acceleration

The data enhancement will involve removing 15% at the start and end of the raw data, since the beginning and end of movement may not be necessary and is inconsistent with the focus of the gait movement. The gait data will go through linear interpolation, which will join the information from the x, y, and z axes,using the equation below:

$$d = d_1 + \frac{g - g_1}{g_2 - g_1}(d_2 - d_1)$$

This linear interpolation is then smoothed. Weighted Moving Average is used for the smoothing of data.

This smoothing of the data is then placed into a histogram by putting the smoothed data into a magnitude vector with the equation below:

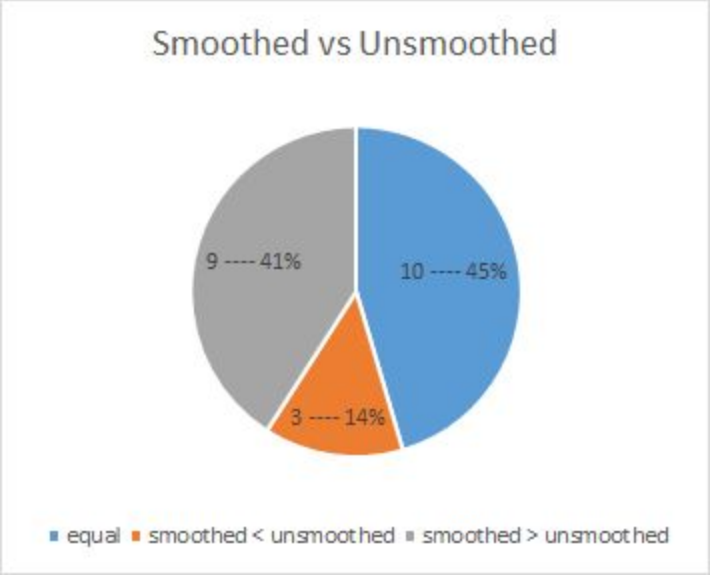
$$m[i] = \sqrt{x[i]^2 + y[i]^2 + z[i]^2}$$

After all 22 subjects have had their data converted to histograms, they are stored in a database.They are later used for matching the template with all the queries from the database. The matching algorithm that was used was the Dynamic Time Warping algorithm.

The Python program is based on taking in the supposed subject's gait and finding other similar matches based on anything with a dynamic time warping distance less than 0.001. Anything with distance of 0.0 is an exact match, but the other similar one's may be considered to simulate differences in intrapersonal gait (since we are only working with one gait walk per subject) as well as interpersonal gait.

We are given the choice to test on smoothed data versus unsmoothed data. Our hypothesis was that matching the smoothed template to smoothed query data (based on the DTW distance stated above) would give more results compared to the matching of unsmoothed template to unsmoothed query data. That is because unsmoothed data supposedly contains more details in the data, so it would be more difficult to match. Whereas, smoothed data has less details, so more matches would occur.

Results



The smoothed data ended up for the majority either having the same amount of matches with 45 percent or more at 41 percent depending on the test subject that was chosen. 14 percent of the unsmoothed data was greater than the unsmoothed data, a small percentage that did not follow our hypothesis.

We originally thought that because the unsmoothed data had a larger amount of data it would result in more matches than the cleaned up smoothed data. This discrepancy may have been due to having to match the histograms which already reduce the amount of data greatly compared to the actual linear interpolation.

Summary

Even though a person's gait is not always going to give the same exact results for authentication every time, it should be similar and consistent. We were only given data of gait cycles of people that walked only one time. Which is why in this project there was a threshold for Dynamic Time Warping with distances below 0.001, to simulate different but similar histograms in order give some room for other similar looking histograms.

We chose to work with matching the histograms of the magnitude vector through Dynamic Time Warping because previously trying to output the Dynamic Time Warping results with the unsmoothed and smoothed linear interpolation was causing the Python program to crash. This may be due to the huge amount of data still in both versions of linear interpolation. This brings the project to another point, which is still being able to

match the template and query without having to read a lot of data in order to reduce costs, time, and noise.

However, since the majority of the results from matching the smoothed and unsmoothed had no major differences, it would be good to warrant a more secure and advanced way of authentication. One of those could be a knowledge based authentication paired with gait, such as storing the user's chosen username and password/pin along with their gait at the beginning of completing a database. In the future, it would be sufficient to include the knowledge based authentication in order to reduce the actual results to the match the person's actual gait. In other words, if the person's gait was entered in for authentication and pulled up other similar results, as long as their password/pin was within those similar results it would warrant a successful authentication. We would also need to lower the threshold more closer to 0.0 to narrow down the results.

Another thing that could have been done differently to make this project better would be to find multiple and average gait cycles of the same person, and use gait cycle detection instead of histograms.

References

- B. Mjaaland, Bendik & Bours, Patrick & Gligoroski, Danilo. (2010). Walk the Walk: Attacking Gait Biometrics by Imitation. 361-380. 10.1007/978-3-642-18178-8_31.
- Lecture 4 Slides

Report 1

Lips Recognition

Passwords are phrases, words, or personal identification numbers (PINs) used for the purpose of securing information. The process of authentication is simple; he who knows the correct password is considered to be a legitimate user and he does not is an imposter. An archaic but effective method in a pre-industrial revolution era, but a terrible method to adapt to a 21st century use case - smartphones. In modern times, the user cannot be completely trusted because users tend to do things the easy way. Users for one cannot be trusted to use safe and cryptographically challenging passwords or be expected to not write them down in form or another [1]. Here the abilities of biometric authentication can be shown to great effect. As biometric authentication relies on who you are i.e. something that cannot practibly be lost or stolen.There exists a variety of biometric authentication modalities, methods, and frameworks here I propose a different one.

My modality will be lips recognition. Specifically speaking I will be elaborating on an authentication method using a combination of voice, facial movement, and vibration. The intent here is firstly to remove the risk of password being stolen and social awkwardness. Since each user will have the same spoken password, that is all users will use the same phrase the same way. The spoken phrase is meant to be taking similarly to an activation phrase for the phone. Meaning saying the password will cause the phone to authentication you and the phone will be listening for your voice. Very

similar to modern Iot devices like the Amazon Alexa which can be prompted to listen with the activation phrase “Alexa” [2]. The voice data will be used in authentication, but will not have as much weight as the next two characteristics. The next of which is captured facial movement using a front-facing camera. The camera will be placed very close to the face while the user recites the password. The captured movement will then be used for authentication. And lastly using the phones movement sensor specifically the magnetometer the slight magnetic field changes caused by motion of the lips. Experiments have shown that a small magnetometer is all that is needed to measure the magnetic field changes from something as small as eyelid movement [3]. Magnet sensing systems (magnetometers) have in the last decade risen to popular use in the medical field for their accuracy and I intent to apply the same effect using the built in sensors of smartphone [4]. The other intent is to be secure, for that a combination of all three will be used in the decision making process.

This modality, has been introduces around the premise that it will be used for authentication on mobile devices. Furthermore, mobile devices being a highly personal object tend to only need authentication, but that is not to say that identification is not also possible. Technically speaking, the system can be enlarged and used enmass for commercial identification in the same way fingerprint is. Identification will however be very off putting to an user, given that they will likely be putting their face very close to a sensor that many people have already used. In light of the expected negative reaction from users, the modality will therefore be confined to authentication only.

The workflow will consist of 4 steps: data capture, feature extraction, matching, and decision. Data capture will happen as described above. Feature extraction will comprise of two steps. First noise reduction must be applied to the magnetometer, mcorphone, and the camera input. Then a machine learning method will extract the pertinent features from each respective input separately. Afterwards matching will take place. The users query will be matched against the preexisting templates. There are multiple templates since the users facial movements and voice can change in response to their level of awakeness. There will be three separate scores produced in the matching phase. Lastly, a decision will be made using the score-level fusion of the three prior scores with more weight given to the magnetometer followed by the facial movement followed by the voice input.

The system is cooperative and user friendly. The user must participate but they are allowed variance at no security cost. This does also mean the system is overt and non supervised given the exclusive and private nature of smartphones.The system is somewhat controlled as specific movements are required for authentication to happen i.e. speaking to and holding the phone in a particular way. Also the user will need some habitation due to the uniqueness of the design as any bold new system mandates.

My modality admiintly is not as universal as some other biometric modalities. People who cannot move their face, or talk cannot use it, but this still does leave the vast majority of the population. Uniqueness is expected to be high because the inherent uniqueness of the face but will require some testing to prove. The modality is expected to be permanent and not affected too much by damage done to the face. Measure

Ability and performance are questionable given the sensitivity of the magnetometer needs to be high and since three scores computed the performance on lower end phones will be low as well [5]. However, the system should prove very difficult to circumvent given that it requires a 3D face with a moving mouth just the same way as a specific human and a voice replay attack to be even approachable to intrusion.

References

1. L. O'Gorman, "Comparing passwords, tokens, and biometrics for user authentication," in *Proceedings of the IEEE*, vol. 91, no. 12, pp. 2021-2040, Dec. 2003. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1246384&isnumber=27928
2. López, Gustavo, Quesada, Luis, Guerrero, Luis, Nunes, Isabel L., Alexa vs. Siri vs. Cortana vs. Google Assistant: A Comparison of Speech-Based Natural User Interfaces. URL: https://link.springer.com/chapter/10.1007/978-3-319-60366-7_23
3. Y. Sonoda, "Applications of magnetometer sensors to observing bio-mechanical movements," in *IEEE Transactions on Magnetics*, vol. 31, no. 2, pp. 1283-1290, March 1995. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=364819&isnumber=8356
4. Linda B.Johnson, SeanSumner, Tina Duong, Posu Yan, Ruzena Bajcsy, R. Ted Abresch, Evande Bie, Jay J.Hana, Validity and reliability of smartphone magnetometer-based goniometer evaluation of shoulder abduction – A pilot study, URL: https://www.sciencedirect.com/science/article/pii/S1356689X15000521
5. M. Shoaib, H. Scholten and P. J. M. Havinga, "Towards Physical Activity Recognition Using Smartphone Sensors," *2013 IEEE 10th International Conference on Ubiquitous Intelligence and Computing and 2013 IEEE 10th International Conference on Autonomic and Trusted Computing*, Vietri sul Mere, 2013, pp. 80-87. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6726194&isnumber=6726171

SPOOFING: REPORT 2

November 28, 2018

Facial Recognition: Can you trust it?

Read More to Find out Now

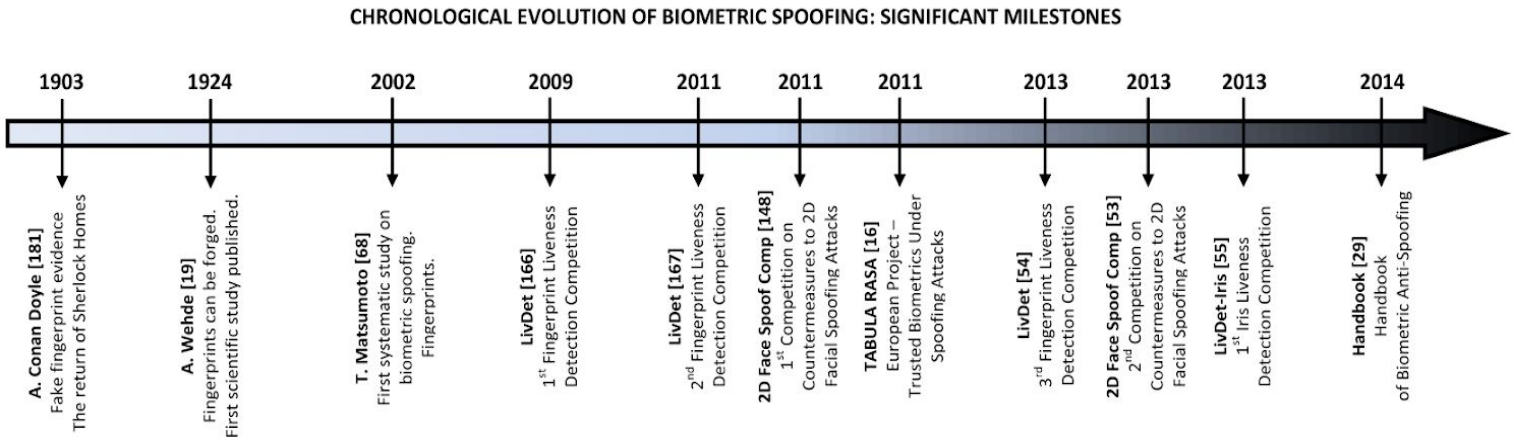
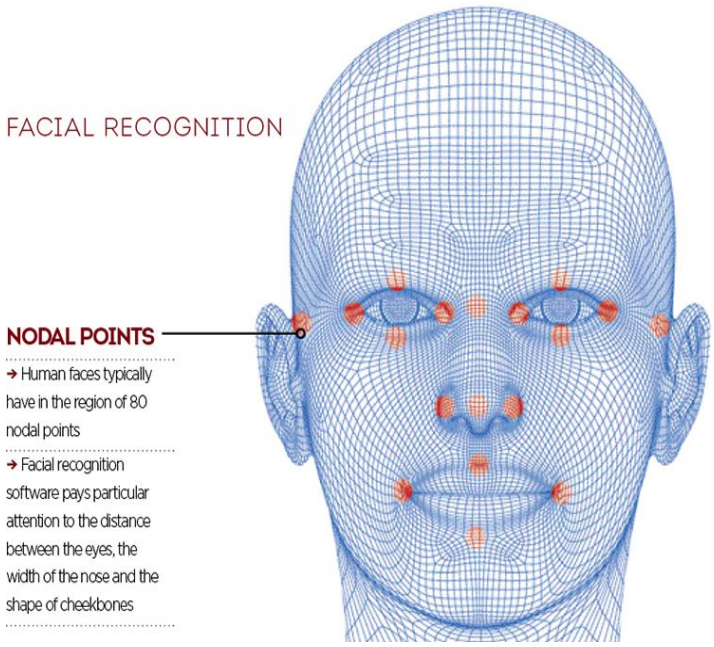


FIGURE 1. **Chronological evolution of biometric spoofing and anti-spoofing with a series of significant milestones that have been covered in this field so far. All events are referenced throughout the article.**

Introduction

Security is a big concern for people and companies in all fields; for we all have secrets to keep. However, in order to do that we need reliable systems in place. Systems that can protect our stuff from malicious attackers.

Biometric Authentication is name for the category of systems that protect our secrets using information we keep on ourselves - our bodies. The specific pieces of out bodies used for the authentication are called modalities. Here we will talk about the face modality (Facial Authentication).



Facial Authentication relies on scanning a person's face for facial contours, pores, and other facial characteristics that make a persons face unique.

Anti-Spoofing

Facial Authentication is in practice a very effective way to differentiate between attackers called imposters and genuine users. However, there's a problem. Any field dedicated to securing something also included the study of methods to break that security.

In Biometric Authentication for any modality this is called 'Spoofing'. When a system is spoofed it has been deliberately made to allow an imposter in. There is a constant arms race between those who make the systems secure and those who break. However, people can be ease for the study of **'Anti-Spoofing'** is all about deflecting these spoofing attacks. Below, there are three ways that Facial Authentication systems are kept safe from attackers.

1) *Challenge Response strategies*

For Facial Authentication to happen you need a sensor which will take a photo of you face to compare against saved photos. From this an obvious spoof arises, tricking the sensor using an existing photo of the genuine user. After all, due to the Internet photos of most people can be found all over social media. To overcome this 'Liveness Detection' is implemented; detection that the subject is alive i.e. not a photo.

There are ways to combat this. One way is to have the sensor look for movements like eye-blinks and gaze changes these are called 'Challenge Response strategies' . These little things are highly effective against photo attacks, but sometimes the imposter will use a video which has all these things. To combat video based spoofs, there are a fair amount of techniques. They are fairly complex but can generally summed up as looking for cues that the information presented to the sensor is not a video by looking at things that a video on a screen produce, or try to take in information from the surroundings that would show the sensor there was no live human in front of it.

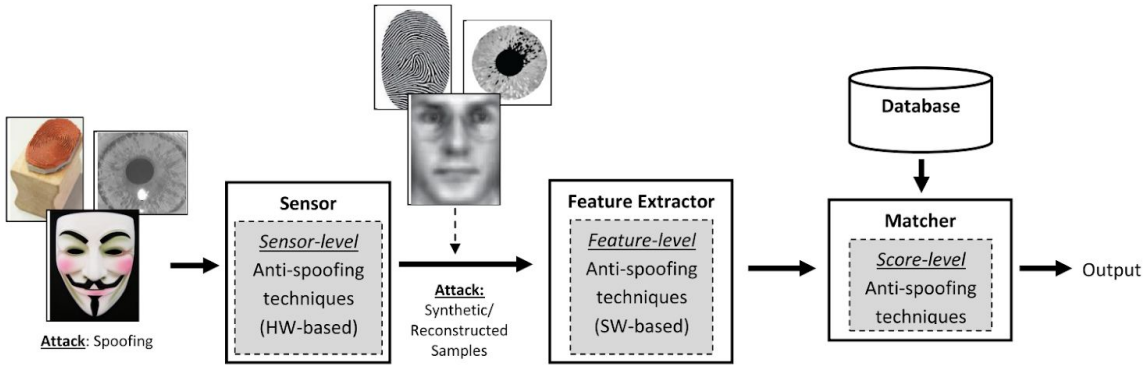


FIGURE 2. **General diagram of a biometric system specifying the modules where the three types of anti-spoofing techniques may be integrated (sensor-level, feature-level and score-level). Also displayed are the two different type of attacks for which anti-spoofing techniques may offer protection: spoofing and attacks carried out with synthetic or reconstructed samples.**

2) *Sensor Modifications*

As described in the last point, the most common form of attack against facial recognition is a video or photo being provided to the sensor. There are a few ways which the sensor itself can be improved without changing the software.

One simple way is to add inferred information to saved genuine information. Humans radiate heat that can be seen with an infrared camera and cannot be easily shown on a video. Moreover even if the imposter naturally looks very close to the genuine user like a twin, this has been shown in [1] to still be effective. Another way is to add lights in a certain arrangement to the sensor. Since a video will be played on a screen it will be reflective, the lights can simply make it very hard to detect anything if a screen is placed in front of it,

3) *Using multiple biometrics*

As simple to way to improve the effectiveness of facial Authentication is simply to use more than one modality. Supplement the reading of facial information with say fingerprint or iris modalities or even voice. This does not completely mitigate the dangers spoofing but certainly raises the bar considerably for an imposter to gain access.

Overall

Facial recognition is a very good form of Authentication, but it still has problems. Problems which come in the form of attacks from imposters, to that tend anti spoofing is there to save it. Regretfully, anti-spoofing does not always work.

There are problems with each of the methods. Challenge Response methods are the best for the device since the sensors do not need to be changed, but have the problem of becoming increasingly complex with time. Checking for blinks and movements slows down the recognition time, and anti-spoofing against videos requires computationally intensive math which drains the battery. Their advantage is also just that, they can be improved without

needing to get a new device. These sort of improvements also fail when the imposters get even better spoofing attacks.

Sensor modification is not very good for an user wallet. Adding any sort of modifications to the device requires money and a new device, but is a better choice long term. Because it sets a much higher bar for an imposter to begin to attack the device. The device as well suffers because of added battery consumption, but everything else should stay the same.

Lastly using multiple biometric would yield the best security at the worst price. It would require a new phone with extensive modifications to allow for a new sensor to be integrated which would be paid for by the users in money and by the device in accommodating all the extra computations and resources needed for it. And like a sensor modification it can still fail as it comes with its own weaknesses like facial recognition does, but sets the bar much higher for an imposter to begin attacking the device.

The Future

No security system is perfect: Biometric Authentication or not. But, we can work towards making a system that is really secure. To that end, Biometric Authentication is far more secure than simple password or pin for unlocking things. Yet, the still do make mistakes. From the research, the future direction of Facial recognition seems to be going towards better sensors and combinations with other sensors in terms of newest growth, but most of the research has up to now been focused on improvement using software.

References

Images

1. Figure 1, From Research article [1], page 1532
2. Picture of Nodal points from The NEW ECONOMY
    - https://www.theneweconomy.com/technology/facial-recognition-is-rapidly-progressing-but-at-what-cost-to-privacy
3. Figure 2, From Research article [1], page 1533
4. Picture of phone unlocking device from DroidViews
    - https://www.droidviews.com/use-facial-recognition-to-unlock-android-devices/

Research Articles

1. J. Galbally, S. Marcel and J. Fierrez, "Biometric Antispoofing Methods: A Survey in Face Recognition," in IEEE Access, vol. 2, pp. 1530-1552, 2014.
   doi: 10.1109/ACCESS.2014.2381273
   URL:
   http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6990726&isnumber=6705689