Gesture and Sociability-based Continuous Authentication on Smart Mobile Devices

Zachary I. Rauen*, Fazel Anjomshoa*, Burak Kantarci†
*Department of Electrical and Computer Engineering, Clarkson University
Potsdam,, NY, USA
†School of Electrical Engineering and Computer Science, University of Ottawa
Ottawa,, ON, Canada
{rauenzi,anjomsm}@clarkson.edu,burak.kantarci@uottawa.ca

ABSTRACT

In this paper, we propose a new continuous verification platform on smart mobile devices. To this end, we integrate gesture-based features with interaction with social networking apps to verify user identities without minimum requirement for a password, pin code or biometric means. The continuous verification subsystem of this work proposes a novel two-step system for verification of users. The subsystem works by having two accurate models working as a primary and backup; when the primary fails the backup takes over to confirm or deny the conclusion of the primary model. The false acceptance rate (FAR) and false rejection rate (FRR) achieved under the proposed two-step system are shown to be 2.54% and 1.98% respectively, compared to the FAR and FRR of single-step verification, which achieved 3.15% and 9.13% respectively. Furthermore, the proposed system also improves the stability of continuous verification. In this work we show that the single step systems are inconsistent when analyzing small feature sets or slightly varied datasets. During both of these instances, the proposed system stays consistent, maintaining a high verification rate.

CCS CONCEPTS

• **Networks → Cyber-physical networks**; **Wireless access networks**; • **Computer systems organization → Sensors and actuators**; • **Computing methodologies → Machine learning**; • **Security and privacy**;

KEYWORDS

behaviometrics; user profiling; machine learning; social networks; gesture recognition

ACM Reference Format:
Zachary I. Rauen*, Fazel Anjomshoa*, Burak Kantarci†. 2018. Gesture and Sociability-based Continuous Authentication on Smart Mobile Devices. In *16th ACM International Symposium on Mobility Management and Wireless Access (MobiWac'18), October 28-November 2, 2018, Montreal, QC, Canada.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3265863.3265873

MobiWac'18, October 28-November 2, 2018, Montreal, QC, Canada

ACM ISBN 978-1-4503-5962-7/18/10…$15.00
https://doi.org/10.1145/3265863.3265873

1 INTRODUCTION

Human-Computer Interaction (HCI) is a mix of human psychology and computing, making software design a science. The field often involves the classification and identification of types of users and their behavior. [23]. By integrating a machine learning component to HCI the identification and classification of a user's behavior becomes a much easier task. Many forms of supervised learning are suitable for usage pattern analysis [3]. Putting this into practice, the authors of [10] study using mobile usage as a means for user identification. In the course of this work they account for many factors, ranging from button presses to actual numbers dialed. The authors test multiple different methods of analysis, including different hash sets and algorithms as well as different feature set choices. Their bloom filter and hash set show promising results in their experiments by reporting that real-time analysis of mobile usage patterns is not only possible but it can be quite accurate. In a work by Anjomshoa et al. [4], the authors create and use a mobile application called TrackMaison ("Track My Social Activity in Social Networks") that is capable of monitoring social media activity and the sensors being used during the activity. Their analysis use a supervised learning mechanism in the form of an SVM as well as an unsupervised mechanism, namely DBSCAN. The results in that research show that it is possible to verify users with less than 10% false rejection rate, and up to 90% accuracy for biometric authentication. An analytic work by Ramadan et al. [22], analyzes touchscreen monitoring and shows that session analysis performs better than raw analysis of touchscreen data. The researchers also found that the more abstract the model of feature set, the more training is needed for machine learning algorithms to be accurate. They recommend training a final product machine learning algorithm on various models, screen sizes and resolutions among other variables.

As mentioned by various and widely accepted sources, smartphones and personalized smart devices combined with effective HCI solutions are strong candidates of non-dedicated sensing systems, commonly known as mobile crowd-sensing systems (MCS). One of the biggest issues in a MCS system is user trustworthiness. In [20], the authors state that the correctness and truthfulness of the acquired data must be verified. This is because a user could be malicious and submit untruthful data, or a device could be malfunctioning and submitting invalid data. This notion is reinforced in multiple works [18, 19, 21]. In both [20] and [18], the authors define new ways of trying to overcome the issue of trustworthiness using scoring and game theory respectively. In the works [21] and [19], the trustworthiness verification systems are applied to applications

through simulations and their results seem to be effective, however below what a real-world application of an MCS should aim for. One way to start overcoming the trustworthiness issues, is to introduce a sociability-based verification system where sociability is the usage of different social media applications by individual users. This is a topic that has already been partially explored. In [5] the authors show a sociability based verification can dramatically improve trustworthiness of the users and their data. This can be improved further by adding in another set of user-specific data, in this case gestures on mobile devices. Having these two sets of data combined can improve the verification process for the users and help overcome the trustworthiness issues.

In this paper, we aim to address this issue by a continuous verification platform that bridges gestures and sociability. The proposed system utilizes two machine learning-based verification models that would complement each other. In this system sociability and gestural features complement each other. In the case of a failure in the verification of a user on the smartphone under one of these models, verification through the other model is called for through a posterior test in order to verify or reject the user profile based on the behaviometric data. On small scale test scenarios, we achieve 99.99% verification success, which is a remarkable improvement over the 93.28% success of a single-model system.

2 BRIDGING GESTURES AND SOCIABILITY IN CONTINUOUS VERIFICATION

A cyber-physical identity (CPI) is a digital representation for a physical known for a specific user. For this work, we can form CPIs to improve continuous verification where the physical known is the touch-screen data yielded from gestures. The previous work involving TrackMaison [4, 6] identified users using sociability (i.e., social activity and sociability factor) which shows it has a similar quality to CPI; users produce unique data. Since both CPI, in the form of gestures, and sociability can be used to identify users, their data can be combined to improve their verification rate. Since the previous sociability verification was done using machine learning (ML) techniques we adopt the sociability-based verification and extend it to improve verification success by introducing gesture tracking into sociability tracking. Machine learning (ML) exploits historical and/or current data to solve problems including those involving prediction of behavior [2]. Therefore ML has been a popular method to address user verification and authentication based on behavioral biometrics [14, 15, 24].

Our approach is to study various ML models and choose one based on the model accuracy. To start, two different ML models are tested for use on the gesture data. They serve as an entry point to understanding the data. Both were built and tested by the scikit-learn package [1]. First, a simple linear logistic regression (LLR) is modeled as seen in Equation (1) [25],

$$P = \frac{1}{1 + e^{-t}} \tag{1}$$

In the equation, t is represented as $\beta_0 + \beta_1 x$ where β_0 and β_1 represent the intercept from the linear regression and the regression coefficient respectively. The goodness of fit is either calculated by the classic R^2 value or the newer Pseudo-R^2s. R^2 is the square of the correlation between the predicted and actual values. This can be represented as

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}. \tag{2}$$

with N representing the size of the data, y being the output, $\bar{y}$ is the average of all output values and $\hat{y}$ is the expected outputs [13].

Similar to R^2, the pseudo-R^2s represents the fitness of the model. There are several versions of these calculations each with their own benefits and failings, but they all center around using R^2 as a base for categorical data such as in machine learning. Since they are variations of R^2 they have often been labeled as pseudo-R^2. Freese and Long present a comprehensive breakdown of the common pseudo-R^2s and their benefits in [13].

Training and execution is performed using the data collected during the first session. This particular model can achieve an accuracy level up to 55%. We chose to test with this model first due to its simple nature; This serves as a good baseline against the RF-based modeling since LLR models have similar performance in terms of speed, and for a system like this a faster analysis is ideal.

We adopt a random forest (RF) model trained and run on the behavioral data. Random forest works on a data set D represented as shown in Equation (3) where x_k is a feature vector of d dimensions (i.e., features) as shown in Equation (4).

$$D = \{(x_1, y_1), ..., (x_n, y_n)\} \tag{3}$$

$$x_k = (x_{k1}, ..., x_{kd}) \tag{4}$$

An RF classifier is a large classification tree with a classification function that outputs a binary value. This makes RF a large binary tree where each decision is based on determining if a given feature, X_i is less than a predetermined threshold, a. In order to build this tree the model must split the data in a half and continue that down the tree until the splitting is no longer significant. The splitting is determined by minimizing the residual sum of squares (RSS) as shown in Equation (5) where y_L and y_R represent the mean y-value for the left and right sides, respectively. Once the tree is built via splitting, it can be used for testing. In an ML application the training done on top of random forest is called bootstrap aggregation. In simple terms the training data is randomly broken up and separate trees are developed for these subsets of data. Predictions and probabilities for unseen samples are then made by averaging the prediction across all the individual trees. This technique is commonly used because it decreases the variance of the model without increasing the bias. It also helps reduce the issues with typical RF classifiers where outliers or noisy data cause erratic results.

$$RSS = \sum_{left}(y_i - y_L)^2 + \sum_{right}(y_i - y_R)^2 \tag{5}$$

In our study, the RF goes through two different tests each having a separate criterion. The first criterion is the entropy values, this yields an accuracy of 94.4%. The second criterion, Gini index (impurity index), leads to an accuracy of 93.4%. Although the difference between the two is small, it is safe to state that the entropy criterion performs better and is thus used going forward. The next stage of

[1] http://scikit-learn.org/

the analysis is to build a similar model for the new sociability data. The model's accuracy performs similarly with a 93.7% accuracy.

Building a model containing separate datasets which are not guaranteed to be correlated is a complicated process. The computation time for training and testing is often increased for these more complex models. Instead, in the interest of time as well as realistic computational limitations, it would be beneficial to leave the two as separate models that work together during the verification.

To this end, in this system the authentication is handled by the primary model until a failure in verification. In that case, the backup ML model is used to double-check the failure of the primary model. With the backup model for this system being proven to identify and verify accurately in the previous works, if the primary model is also accurate the overall system becomes very accurate. Specifically in this work, we test both gestures as primary and sociability as primary.

To build this system, an ML model type needs to be chosen. It is worth noting that various ML algorithms can be used in this system; however the purpose of this study is to show the applicability of a two-stage ML-based approach in continuous verification of smartphone users. To this end, one candidate is random forest (RF) classifier. RF performs very efficiently and accurately when encountering data similar to what it was trained on, and in our testing outperformed the LLR classifier. One of its main drawbacks is that the accuracy falters significantly when encountering data that does not follow any of its predetermined patterns or could be considered an outlier. However, this work is focusing on increasing accuracy of verifying already known users without introducing spoofing into the system. This also makes finding a way to improve to spoof-protection of RF into a possible future work. In the next section, we also include an analytic justification for the selection of RF as the ML model for both sociability and gesture datasets.

3 PERFORMANCE EVALUATION

The feasibility study of the proposed two-stage verification has been performed with a limited participant set of users across two different sessions. Each of these sessions used a different tracking service (i.e., TrackMaison 1.0 and TrackMaison 2.0). In the first session which ran for five weeks, the participants used the energy efficient service in TrackMaison 2.0 [1]. In the second session, the non-optimized service in TrackMaison 1.0 was used for only two weeks. This shortened session is only to serve as an additional control for comparison between usability of the two apps. The previous version of TrackMaison was done with this service so there is already sufficient data. The collected data is provided by NEXTCON Research Laboratory and can be found at [17].

3.1 Feature Set and Its Reduction

In addition there is an entirely new set of data being added in this work. The new data comes from the gestures performed while using the TrackMaison application. The data that can obtained from gesture tracking varies from gesture to gesture since some gestures have multiple events. One example of this is a double tap, since there are two touches in a double tap there is twice the data of a single tap. The gestures being tracked in this work are the following: single press, long press, double press, fling, and scroll. The default case is that the gesture contains one event, an x location, a y location, and the type. The last two events–fling and scroll–both have two events and a different set of initial data. Instead of location, they have velocity and distance respectively. Each event is comprised of its own set of data. This includes the following: action, actionButton, id, x, y, toolType, buttonState, metaState, flags, edgeFlags, pointerCount, historySize, eventTime, downTime, deviceId, and source.

The correlation matrix has been obtained using the 'pandas' and 'seaborn' packages in a python environment. The 'pandas' package is used for the analysis as it has become a standard in the python environment. 'Seaborn' is a package that helps with the visualization of the data by creating a heatmap for the matrix. The result of correlation analysis is illustrated in Figure 1.

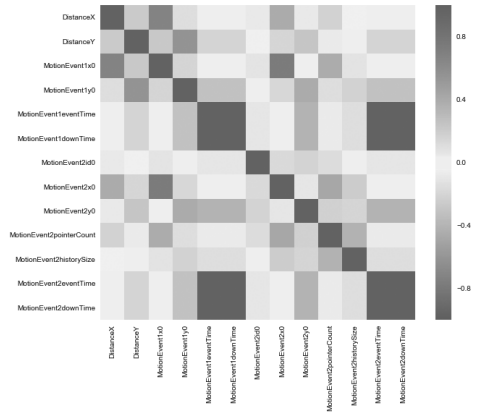

Figure 1: Correlation matrix for gesture data.

The high correlation along the $y = -x$ axis of the figure shows the correlation of each feature to itself, which is intuitive that all the values on the diagonal are 1. Some other highly correlated features include MotionEvent1eventTime and MotionEvent1downTime. This makes sense because on the Android platform for the first motion event in a gesture, these two values are always the same hence the value of 1 on the figure.

Another noteable relationship is the higher correlation between MotionEvent2x0 and MotionEvent1x0. The high correlation between these features can be explained by fling and scroll gestures being the only ones with two events. When users want to fling through or scroll a list they tend to do so in a vertical manner restraining their movement in the x direction.

Although feature set reduction has been done in previous works for the sociability data the new battery related features could have correlations that were not known. To rectify this, the sociability data went through the same process as the gesture data resulting in the correlation matrix seen in Figure 2.

As can be seen in the figure, there are several highly correlated features. *Timestamp*, *timestart*, and *timeend* are highly correlated with one another, therefore two of them should be removed leaving *timestart*. Furthermore, *batteryStart* and *batteryEnd* are also correlated as this can occur due to the battery levels not changing much when opening and closing a social application. BatteryDrain

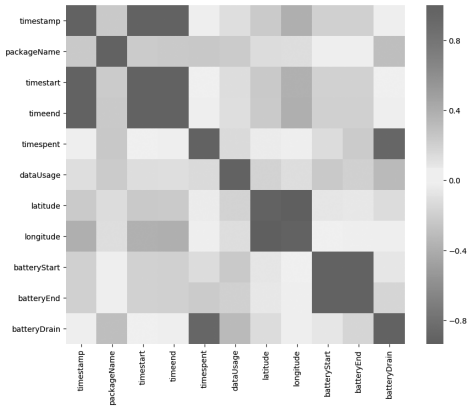

Figure 2: Correlation matrix for sociability data.

is highly correlated with *timespent*, and slightly correlated with *datausage*. While correlations under both cases are intuitive for this system, since *batteryDrain* and *timespent* are both highly correlated so one must be removed. In this case, it is more viable to remove *batteryDrain* since it has a higher overall correlation with the rest of the features, mostly due to its slight correlation with *datausage*.

3.2 Results under Selected ML Model

To determine the threshold in the RF algorithm, the threshold should be varied and compared to the overall performance of the system. Like any other test this should be averaged over several runs, in the case of this work it was repeated and averaged 500 times. Verification here and for this work is the accuracy with which users were properly authenticated as themselves. The result of this test can be seen in Figure 3.

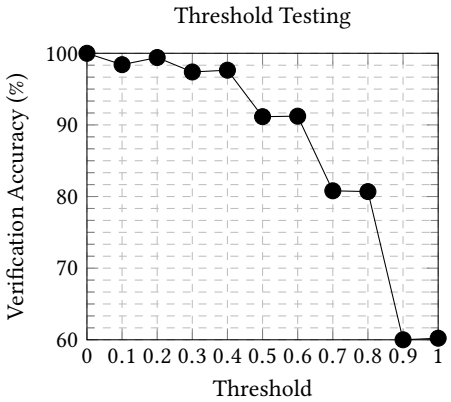

Figure 3: Performance of various methods on the first collection session.

For this system 0.6 allowed for very high performance and accuracy and should still deny most spoofing attempts in future works. This is because when the genuine user had a low probability such as 0.6 while the other users only reached a maximum of 0.4 which leaves significant room. Additionally according to the graph, 0.6 is the highest threshold that can be used before significantly performance impacts are seen; dropping from 91.22% to 80.80% between 0.6 and 0.7. It is also significant in itself that 0.6 happens to be the highest thresholds to be above 90% which is generally the bare minimum to look for in these tests.

Upon determining the threshold, in order to ensure stable and reliable results the algorithm ran on each dataset (gestures and sociability) 1000 times with the final result being the average performance. For the first collection session the accuracy for gestures and sociability were 92.45% and 98.12% respectively. The results for the second session were 93.25% and 99.52% for gestures and sociability respectively. The results for the sociability data are skewed high due to having a smaller dataset which does not allow for as much testing, as well as have a larger number of features than the gesture data.

To complete the two-model analysis the previous methods need to be altered to have a fall-back. In this methodology, during a failing test case for the primary model, the most closely related data is found and tested by the secondary model. If both models fail then it is considered a fail, otherwise either model is able to verify.

For the gesture and sociability data, the close relation is determined by the smallest difference in event times between the two sets. That is, if a gesture fails to verify, the sociability data with the closest timestamp for that device is used to verify. Both possibilities of this model are tested; gestures and sociability switching off as the primary model. For the gestures-sociability (G-S) model, gesture-based verification serves as the primary model whereas in the sociability-gestures (S-G) model, sociability serves as the primary model.

In the G-S model, the overall accuracy in the two sessions end up being 99.25% and 99.99% respectively. The mirrored model, S-G perform similarly yielding 99.99% and 99.81% for the respective two sessions. From these overall results one would refrain from favoring one particular version of the dual-model system, however both S-G and G-S outperform the single-model analyses.

An interesting way to view the performance is on a user-by-user basis. Not only does this give a unique perspective on the performance of the algorithm, but it can help uncover deeper insights into the data itself. It is to be expected that the results for the second session are less consistent than the first session due to the smaller dataset size. This limits not only the amount of data that can be tested on but the amount of data that can be trained with. Here, we go through a selection of users to show different performances. We start with the anonymous user *A* in Figure 4.

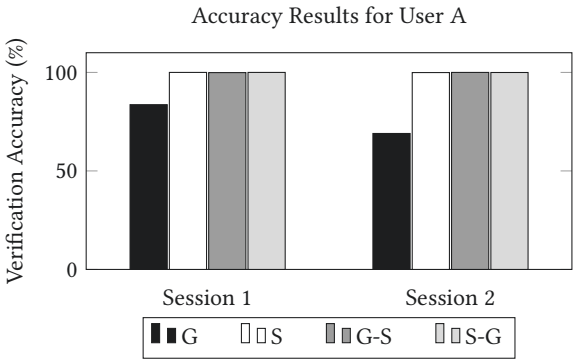

Figure 4: Performance of various methods for User A.

For all the user-by-user charts, the labels on the x-axis represent the different analyses run on the data. *G* represents the gesture-only model, *S* represents the sociability-only model, $G - S$ represents the dual model approach with gestures acting as the primary and sociability as the backup, and $S - G$ represents the flipped version of that where sociability is the primary model backed up by gestures.

Looking at the gesture-only model, one can see the accuracy varies quite heavily, this is due to having a small feature set. If the anomalies were eliminated and those additional features added back in the performance would likely be a lot more consistent.

The results for User *B* depict a similar situation, except in this case both the single model analyses vary in accuracy. As can be seen in Figure 5.

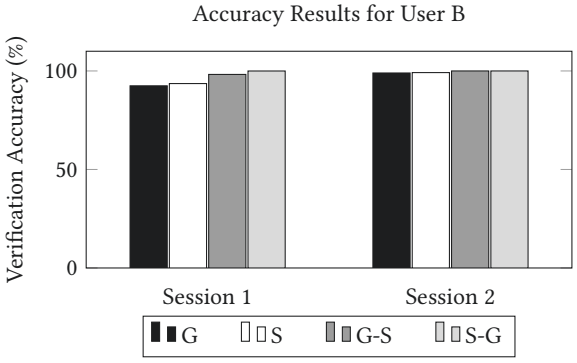

Figure 5: Performance of various methods for User B.

The single model analyses for User *C* also have a slightly larger variation than the dual model analyses. It can also be seen that overall the dual model systems yield higher accuracy in tandem with the consistency. This can be seen in Figure 6.

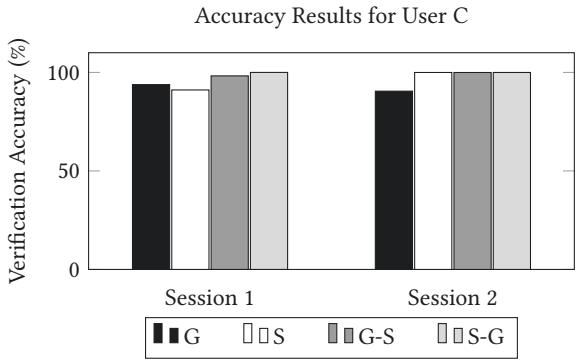

Figure 6: Performance of various methods for User C.

Overall, user *C* performs worse when using the gesture data as primary. That is, the gestures-only model perform worse overall than the sociability-only model and the G-S model perform slightly worse than the sociability-gestures model. This can indicate that this user is a social media user that uses media more heavily. This would mean that they have more sociability data since they are using the application, but in the case of media like video they would not be directly interacting with the application itself.

The final user, User *D*, is the one with the anomalous data so the results for this user will likely be worse than the others. At a quick glance of Figure 7, one can see that user's second session perform significantly worse than the first session.

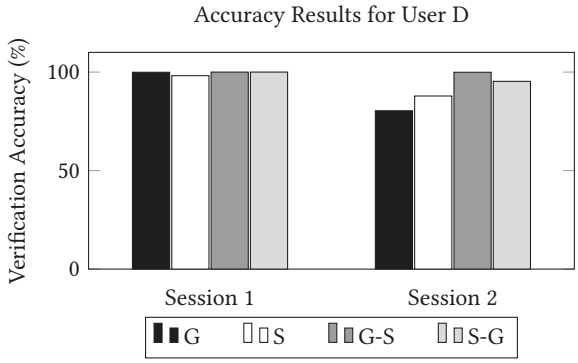

Figure 7: Performance of various methods for User D.

Although this user has anomalous data, the final results for the first session were all very high. This could indicate that the anomalous data from the user is more prevalent in the second session. Here we also see the largest variation from either of the dual-model systems by the sociability-gestures model; dipping from 99.91% to 95.3%. However, this is still more consistent for the user than either single model system by a significant amount. The single model system dropped by 10.32 per cent points at minimum versus the dual model's 4.61.

Overall for the two single model analyses, the accuracy dips as low as 69.01%. There is a similarity between the graphs for both these in that User *A* still performs quite poorly for the gesture only

model. We can also see that although the two single model analyses are unstable with the smaller dataset, from gestures and the second session overall, the dual-model system stays reliable and consistent. The lowest dip between these two models is down to 95.3% for User *D*. The user has an unusually low number of sociability data points during the second session which could lead to a less reliable model. Significant results like these are tabulated in Table 1.

Table 1: User verification testing results.

Model	Average Accuracy	Lowest Accuracy	Largest Variation
G	88.57%	69.01%	19.56%
S	96.22%	87.85%	10.32%
G-S	99.68%	98.24%	1.76%
S-G	99.62%	95.3%	4.61%

From the table, it can be seen that the single model systems performed worse on average, had worse lowest accuracies, and had a large variation between the two sessions. On the other hand, the two model systems performed well in each category with system $G-S$ slightly outperforming the other. In a real world system, $G-S$ would be an ideal model due to the minimal variation between sessions.

In the previous work involving TrackMaison 1.0 [7], the verification tests yielded an average accuracy of 95.56% for sociability data alone. The performance of their model closely resembles that of model *S* as they use the same type of data. Versus the two dual model analyses however, the new dual model systems outperform the old system. Previously, the authors faced issues properly verifying users that fell into different categories of amount of data produced ranging from 90% to 100% accuracy. This is quite similar to how *S* performed across the two sessions of varied sizes. The dual model systems improve upon this aspect as well performing consistently regardless of the dataset size.

In another similar work, authors Meng, Wong, Schlegal and Kwok [16] use a gesture-based verification system to verify users without biometric means. Their maximum reported accuracy was 92.2% which slightly outperforms the model *G*. When compared with the gestures-primary dual model $G-S$, the previous work's accuracy falls short; the dual model systems beat the previous work by $\approx 7.5\%$ percentage points.

The dual-model system has shown that not only can it boost accuracy over the two single model system like in the first collection session, it can also perform stably when faced with a reduced size data set like in the second collection session. Both of these improvements can help significantly in continuous verification as these systems need higher accuracy and want to be able to do it quickly with as little data as possible.

3.3 Cross-Validation Results

In the previous section we explored pure verification where the users were tested only against themselves. In cross-validation, the data from every user is tested against every user to determine which data would be falsely accepted or falsely rejected by their model. The metric to test each model on are called false acceptance rate (FAR) and false rejection rate (FRR). FAR is the rate at which impostors are incorrectly accepted as someone else as shown in Eq. 6. FRR is the rate at which the genuine users are rejected as themselves as seen in Eq. 7.

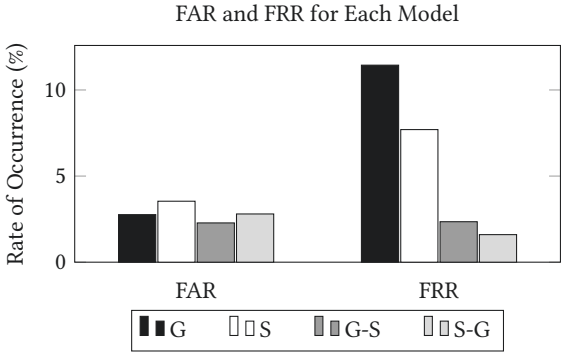

Figure 8: Results of the cross-validation testing.

$$FAR = \frac{impostors\ accepted}{\#\ of\ impostor\ tests} \tag{6}$$

$$FRR = \frac{genuine\ users\ rejected}{\#\ of\ genuine\ tests} \tag{7}$$

Because FRR is testing users against themselves, the results will be quite similar to the previous section however this metric will also be tested and shown here.

The testing setup is the same as the previous section, the only difference is that each data point will be tested against all users as opposed to testing exclusively that user. Here when testing a user's data against itself that is referred to as a genuine test. When it is tested against another user's data it is referred to as an impostor test. The tests here, like in the previous section are the average result of 1000. In these tests the goal is to have the lowest score for both FAR and FRR.

For the resulting values we average the results from each user across both sessions to arrive at an average FAR and FRR. The results for the cross-validation are visualized here in Fig. 8.

From a glance at the figure, both the single model analyses *G* and *S* have significantly higher FRRs than their dual model counterparts by at least 4%. For FRR we can see the best result is $S-G$ barely beating model $G-S$ On the side of FAR, we can see model $G-S$ performs better than the other three models with model *G* and $S-G$ being about equal.

Using Table 2, we can attach more solid numbers to each model by viewing their performance between the two sessions.

Table 2: Cross-validation results across each session.

	Session 1		Session 2	
Model	FAR	FRR	FAR	FRR
G	2.62%	7%	2.91%	15.87%
S	0.99%	10.17%	6.1%	5.23%
G-S	1.35%	3.7%	3.2%	0.99%
S-G	0.15%	1.9%	5.46%	2.01%

From the table we can see the FAR of model G was better than model $S-G$ on average however model $S-G$ significantly outperformed G in FRR. Model $G-S$ performed better than both single model analyses for both FAR and FAR and was only slightly beaten in FRR by the other dual model $S-G$. From these results and the results from the previous section we can safely say the dual model analyses outperform the single model analyses presented here.

To get a sense of how consistent each model performs, we can view how each model performed for each user across both sessions. This is listed in Table 3.

Looking at the results for G and S, we can see a wide variability in their performance between each user. Some users perform really well while others vary by over 10%. We can see here, like before, the two single model analyses perform poorly for FRR with multiple results over 10% for each model.

Moving on to the dual model systems we can see that they maintain consistent performance across all users. Although between the two $S-G$ has more variation of results between users, it still has better results for FRR than $G-S$. $G-S$ on the other hand has a lower FAR which is an important metric in these systems. For a real world application, both the dual models have shown they could perform well, with $G-S$ having the edge due to its better FAR performance.

It is also important to verify results other existing works. Zheng et al. used raw touchscreen data for verification [27]. In their testing they also performed cross-validation testing. Their final result was an equal error rate of 3.65%. Compared to this work we can see that model S performed around the same for FAR but was significantly worse for FRR. Model G performed better for FAR but like model S was significantly worse for FRR. In contrast, both the dual model systems outperformed that work in both FAR and FRR.

Not only were the dual model systems able to accurately verify the users in the previous section, in this section they were able to accurately deny other users from being verified as another user.

4 RELATED WORK

Anjomshoa et al. introduce and develop a social network tracking mobile crowdsensing system named TrackMaison (**Track M**y **A**ctivity In **So**cial **N**etworks) [5]. This was able to track data usage, location, usage frequency, and session duration of five different social network services namely, Facebook, Twitter, Skype, LinkedIn, and WhatsApp. The authors also introduce a social activity rate, the data usage rate of the user and sociability factor metrics, a function of the user's relative session durations. This gets broken down into three behavior types: highly active, moderately active, and low active. Using this setup, their initial analysis was able to show that the system could function as a form of continuous verification for mobile users. Furthermore, they present a relevant case study about Instagram users. In this case study, it is broken down much the same way with the three types of users. The results here are also very promising. The low active users are identified with a negligible false acceptance rate and the highly active users are identified with a false acceptance as low as 3%.

Research using TrackMaison was expanded on this in another work [7] where the researchers collected a large amount of data and did a full experiment with their own framework TrackMaison. The authors looked into using the data with machine learning to be able to identify the users. More specifically they looked into using SVM as well as DBSCAN in verification. In that research, they collected data from around a dozen participants for over two months. They showed that slightly more than 90% of the time an active user can be identified without any need for biometric authentication.

There are many potential fields for applying continuous verification where the benefits would be significant. One of the most direct applications of continuous verification would be in cybersecurity. In [9], Ceccarelli et al. explore applying continuous user verification to cybersecurity, but more specifically the security of Internet services. The authors devise a large-scale high security Internet service to test with both mobile devices as well as desktop computers. The proposal is based primarily of the newer session establishment protocol of using biometric systems over a typical username and password login. The protocol is considered sufficient as a replacement over the typical login, however both succumb to the downfall of being a single verification where the entire session is considered to be of the same user. The authors state this as a downfall for multiple reasons: In the real world, a device can be compromised during usage, and people may use each others' devices which would be a change in user.

In a similar work by Bu et al. [8], the authors explored how continuous verification can help decrease the attacks on mobile ad-hoc networks (MANETs). This proves to be a more difficult task than the internet service from before because in an ad-hoc network, the sensors and biosensors that are accessible through mobile devices are not guaranteed to be consistent.

Continuous verification and identification of users through HCI have been shown perform well in a work by Vural et al [26] where the authors used the time difference between keystrokes in order to profile the users and test their data. The study was able to achieve a False Acceptance Rate (FAR) of 3.45% and a False Rejection Rate (FRR) of 8.82%. These results are quite accurate and are close to what were seen in other works with a similar idea such as the work by Leggett [12]. Both these works adopt a similar principle, and they both test the incoming data based on an expected distribution and known values for each user. This works very well for their test conditions since these works are not currently in a widespread use-case such as MCS.

In [11], the impact of data size on the accuracy of continuous identification was explored. This is a very important proportionality to explore for HCI in the realm of continuous verification since the amount of incoming data from the users will not always be constant. The authors showed that even with a smaller set of testing data, the results for both FAR and FRR can be quite accurate. This result is promising for an HCI-based continuous verification system as small usages in the HCI application in the real world can only produce minimal data sets. The more interesting of the results is the following: If the reference profile for a user is large, the testing data can be fairly small but is still able to produce accurate results.

5 CONCLUSION

We have proposed a dual-model system that relies on gestures and sociability. The dual model systems performed very well boasting an average accuracy of 99.68% for gestures-primary and 99.62%

Table 3: Average user results from cross-validation.

	User 1		User 2		User 3		User 4	
Model	FAR	FRR	FAR	FRR	FAR	FRR	FAR	FRR
G	0.25%	24.72%	6.67%	3.83%	3.78%	6.4%	0.35%	10.78%
S	3.06%	0.56%	4.77%	10.43%	3.74%	8.42%	2.6%	11.39%
G-S	1.7%	2.6%	1.74%	1.91%	3.02%	2.5%	2.55%	2.37%
S-G	2.84%	3.28%	3.03%	0%	2.2%	0.53%	3.16%	2.61%

for sociability-primary. This is an improvement over both single-models having accuracies of 88.57% and 96.22% for gesture and sociability data respectively. Not only did the dual model system improve the overall accuracy, but it was also able to stabilize the verification across all the users. The single-model system was not consistent between sessions or even between the two datasets. Lastly, the new system was also able to maintain its accuracy and stability when used in the second session which has significantly less data and would be considered a small dataset.

During cross validation tests, the dual model systems significantly outperformed the single model systems. They allowed less impostors to be verified as another user, which is an important metric when working with these systems. They were also able to properly authenticate genuine users more often than the single model systems. Quantitatively, the single model systems had an average of 3.15% and 9.13% for FAR and FRR respectively. The dual model systems averaged 2.54% for FAR and 1.98% for FRR.

For both verification and cross-validation testing, the two single model systems shown here performed about the same as systems shown in related works. On the other hand, the two dual model systems outperformed these same systems in both rounds of testing. Between the two dual model systems the gestures-primary model $G - S$ performed better than $S - G$ in most metrics while staying close in others.

ACKNOWLEDGMENTS

This material is based upon works supported by the Center for Identification Technology (CITeR); U.S. National Science Foundation (NSF) under Grants IIP-1068055 and CNS1464273; and Natural Sciences and Engineering Research Council of Canada (NSERC) DISCOVERY Program under Grant RGPIN/2017-04032.

REFERENCES

[1] 2018. Z. I. Rauen, "Improving the role of Human-Computer Interaction in Continuous User Verification in Smartphone Sensing. MSc Thesis, Clarkson University. (2018).
[2] Ethem Alpaydin. 2014. *Introduction to Machine Learning*. ISBN: 978-0-262-028189, MIT Press.
[3] K. Altun. 2012. Machine learning methods for human-computer interaction [tutorial]. In *IEEE Haptics Symposium (HAPTICS)*. 1–3. https://doi.org/10.1109/HAPTIC.2012.6183851
[4] F. Anjomshoa, M. Aloqaily, B. Kantarci, M. Erol-Kantarci, and S. Schuckers. 2017. Social Behaviometrics for Personalized Devices in the Internet of Things Era. *IEEE Access* 5 (2017), 12199–12213.
[5] F. Anjomshoa, M. Catalfamo, D. Hecker, N. Helgeland, A. Rasch, B. Kantarci, M. Erol-Kantarci, and S. Schuckers. 2016. Mobile behaviometric framework for sociability assessment and identification of smartphone users. In *IEEE Symp. on Computers and Communication (ISCC)*. 1084–1089.
[6] F. Anjomshoa and B. Kantarci. 2018. SOBER-MCS: Sociability-Oriented and Battery Efficient Recruitment for Mobile Crowd-Sensing. *Sensors* 18, 5 (2018), 1593.
[7] F. Anjomshoa, B. Kantarci, M. Erol-Kantarci, and S. Schuckers. 2016. A mobile platform for sociability-based continuous identification. In *2016 IEEE 21st International Workshop on Computer Aided Modelling and Design of Communication Links and Networks (CAMAD)*. 149–151.
[8] S. Bu, F. R. Yu, X. P. Liu, and H. Tang. 2011. Structural Results for Combined Continuous User Authentication and Intrusion Detection in High Security Mobile Ad-Hoc Networks. *IEEE Trans. on Wireless Communications* 10, 9 (Sep. 2011), 3064–3073.
[9] A. Ceccarelli, L. Montecchi, F. Brancati, P. Lollini, A. Marguglio, and A. Bondavalli. 2015. Continuous and Transparent User Identity Verification for Secure Internet Services. *IEEE Trans. on Dependable and Secure Computing* 12, 3 (2015), 270–283.
[10] Paul Giura, Ilona Murynets, Roger Piqueras Jover, and Yevgeniy Vahlis. [n. d.]. Is It Really You?: User Identification via Adaptive Behavior Fingerprinting. In *ACM Conf. on Data and Application Security and Privacy*.
[11] J. Huang, D. Hou, S. Schuckers, and Z. Hou. 2015. Effect of data size on performance of free-text keystroke authentication. In *IEEE International Conference on Identity, Security and Behavior Analysis (ISBA 2015)*. 1–7.
[12] John Leggett, Glen Williams, Mark Usnick, and Mike Longnecker. 1991. Dynamic identity verification via keystroke characteristics. *International Journal of Man-Machine Studies* 35, 6 (1991), 859 – 870.
[13] J Scott Long and Jeremy Freese. 2006. *Regression models for categorical dependent variables using Stata*. Stata press.
[14] P. A.K. Lorimer, V. M-F. Diec, and B. Kantarci. 2018. COVERS-UP: Collaborative Verification of Smart User Profiles for social sustainability of smart cities. *Sustainable Cities and Society* 38 (2018), 348 – 358.
[15] L. Lu and Y. Liu. 2015. Safeguard: User Reauthentication on Smartphones via Behavioral Biometrics. *IEEE Trans. Computational Social Sys.* 2, 3 (Sep. 2015), 53–64.
[16] Y. Meng, D. S. Wong, R. Schlegel, et al. 2012. Touch gestures based biometric authentication scheme for touchscreen mobile phones. In *International Conference on Information Security and Cryptology*. Springer, 331–350.
[17] NEXTCON. 2018. Dual Dataset. (2018). http://nextconlab.academy/CITER2/DualDataset.zip
[18] M. Pouryazdan, C. Fiandrino, B. Kantarci, D. Kliazovich, T. Soyata, and P. Bouvry. 2016. Game-Theoretic Recruitment of Sensing Service Providers for Trustworthy Cloud-Centric Internet-of-Things (IoT) Applications. In *IEEE Global Communications Conference (GLOBECOM) Workshops*.
[19] M. Pouryazdan and B. Kantarci. 2016. The smart citizen factor in trustworthy smart city crowdsensing. *IT Professional* 18, 4 (2016), 26–33.
[20] M. Pouryazdan, B. Kantarci, T. Soyata, L. Foschini, and H. Song. 2017. Quantifying User Reputation Scores, Data Trustworthiness, and User Incentives in Mobile Crowd-Sensing. *IEEE Access* 5 (2017), 1382–1397.
[21] M. Pouryazdan, B. Kantarci, T. Soyata, and H. Song. 2016. Anchor-assisted and vote-based trustworthiness assurance in smart city crowdsensing. *IEEE Access* 4 (2016), 529–541.
[22] A. Ramadan, H. Hemeda, and A. Sarhan. 2017. Touch-input based continuous authentication using gesture-level and session-level features. In *8th IEEE Information Tech., Electronics and Mobile Communication Conf. (IEMCON)*. 222–229.
[23] Mike Sharples. 1996. Chapter 10 - Human-Computer Interaction. In *Artificial Intelligence*, Margaret A. Boden (Ed.). Academic Press, San Diego, 293 – 323.
[24] C. Shen, Z. Cai, X. Liu, X. Guan, and R. A. Maxion. 2016. MouseIdentity: Modeling Mouse-Interaction Behavior for a User Verification System. *IEEE Transactions on Human-Machine Systems* 46, 5 (Oct 2016), 734–748.
[25] Barbara G Tabachnick and Linda S Fidell. 2007. *Using multivariate statistics*. Allyn & Bacon/Pearson Education.
[26] E. Vural, J. Huang, D. Hou, and S. Schuckers. 2014. Shared research dataset to support development of keystroke authentication. In *IEEE International Joint Conference on Biometrics*. 1–8.
[27] N. Zheng, K. Bai, H. Huang, and H. Wang. 2014. You are how you touch: User verification on smartphones via tapping behaviors. In *IEEE Intl. Conf. on Network Protocols (ICNP)*. IEEE, 221–232.