A combined kNN-SVM confidence-based algorithm for Continuous Implicit Authentication

Sokratis Koseoglou, Thomas Karanikiotis, Michail D. Papamichail, Andreas L. Symeonidis
Electrical and Computer Engineering Department
Aristotle University of Thessaloniki
Thessaloniki, Greece
sokrkose@ece.auth.gr, {thomas.karanikiotis, mpapamic}@issel.ee.auth.gr, asymeon@eng.auth.gr

***Abstract*—Due to the dramatic increase in popularity of smartphones in the past decade, the need for sophisticated patterns that provide user authentication and security of sensitive information is growing. Towards this goal, a variety of biometrics has been applied in the past few years, such as user password, face recognition or fingerprint in order to access the device. In this work we propose a different approach. A continuous implicit authentication model that recognizes the authenticated user in real time, while using their device, through their swiping behavior. To meet this target, a one-class kNN-SVM confidence-based model was created and tested in a quite representative dataset which consists of 975 different users.**

Keywords*—Continuous Implicit Authentication(CIA);k-Nearest Neighbors(kNN); Support Vector Machine(SVM);

I. Introduction

Nowadays, smartphones are devices which are not used just for calling and texting each other. They are used for managing a lot of daily activities such sending or receiving emails, shopping, mobile payments etc. This can be confirmed, by the recent statistics shown in [1], which show that from 2007 to 2021 there have been sold about 1.56 billion smartphones and we can also see that the annual sold smartphones graph had an exponential form in the past few years. Thus, the need for sophisticated ways to protect these sensitive information contained in our smartphones is of high importance. In the past few years, a variety of ways of protecting our smartphones has been introduced, such as password , Iris scanning [2], face recognition [3]. Those biometrics have had a big success in the goal of protecting user's information from other people in case they lose their smartphone or they get robed. However, they have a disadvantage, which is that the user has to unlock the device every time they want to use it. In [4], we can see that the average user checks, thus unlocks, their phone about 110 times per day. So, the users, in order to save time, sometimes, use simple passwords such as "1234" or "pass" which can be easily penetrated. In that context, it has been introduced the continuous implicit authentication method which comes to solve the problem of unlocking the phone every time the user wants to use it and also ensures that user's data cannot be accessed from an unauthorized user.

More precisely, through Continuous Implicit Authentication (CIA) we are able to develop models that monitor and understand the way that the authorized user interacts with their smartphone, through their swiping behavior, and that way, the models can evaluate, in real time, if the person using the smartphone is an authorized one or not. We can see some recent works that have been focusing on developing sophisticated artificial intelligence algorithms for constructing continuous implicit authentication models for smartphones, in [5][6][7]. In this work, we propose a one-class kNN-SVM combination confidence-based classifier which classifies the user as an authorized one or not, based on their gesture's data. The key difference between this work and the previous ones is that our model is being trained and tested in real world dataset. One main problem of CIA is that the model is consuming computing resources, and thus battery power, each time the user interacts with their phone, so the complexity of the algorithm being used should be taken into consideration.

The rest of this paper is organized as follows. Section II shows the Research Overview and we analyze the dataset used for our modeling, the metrics used for the evaluation of our model and some research questions we tried to answer. Section III presents the whole system, the data preprocessing that we did before training our model and the presentation of our proposed model, while Section IV presents the evaluation results of our modeling. Last but not least, Section V shows some summarized conclusions and some future works that could be done in order to further improve the proposed algorithm.

II. Research Overview

As already noted, CIA uses data from the authorized user's gestures in order to train the proposed model. Then, the model will be able to detect, through gestures, if the person using the smartphone is a legitimate user or not. In case, the model classifies the user as an authorized one, the device keeps working without interrupting the user but if the model classifies the person using the phone as an unauthorized one ("attacker") the smart phone is being locked. So, we can easily understand that, in order for our model to have a high accuracy, the dataset which will be used to train and test our model should be quite representative. In that context, we used a dataset which was created from a public application called "BrainRun" [8]. This way, we managed to create a big dataset which consists of 2221 users that downloaded and used the application. Another asset of this dataset except for the satisfying number of users is that the application is available to many individuals of all ages, genders and level of expertise, creating an unbiased dataset.

In the goal of creating this representative dataset, Papamichail et al. [8] developed "BrainRun" which, as we said before, is a commercially available application and consists of a number of games aiming at boosting cognitive skills of the users. More precisely, BrainRun consists of 5 different games ("Focus", "Mathisis", "Memoria", "Reacton", "Speedy"). Each game has a different set of rules and it is designed in a way that can provide us with a different set of user gestures such as horizontal swipes, vertical swipes, taps etc. In Table 1 we can see the statistics of the dataset created from the application "BrainRun". Despite the fact that the dataset contains both taps and swipes of different users, in this work we only use the swipes of each user, both for simplicity reasons of the training procedure of the model and because the swiping behavior of a user gives us more information about the way they interface with their smartphone. So, we only use the data derived from the games "Focus" and "Mathisis" which give us vertical and horizontal swipes of each user respectively. Finally, the total number of users who played "Focus" or/and "Mathisis" is 975 users which are being used for the training and testing of our model. Then, after collecting the swipes from these users, we calculated some features in order to better understand the swiping behavior of each user. These features are shown in Table 2.

TABLE 1

Metric	Value
Number of users	2,221
Number of devices	2,418
Number of games players	106,805
Total number of gestures	3,110,101
Number of taps	2,463,115
Number of swipes	646,986

TABLE 2

Feature Name	Feature Description
Horizontal Trace Length	Calculated distance between first and last point of swipe in horizontal axis
Vertical Trace Length	Calculated distance between first and last point of swipe in vertical axis
Slope	The slope of the straight line that best approaches the swipe's trace
Mean Squared Error	The mean squared error between the swipe's points and the straight line
Mean Absolute Error	The mean absolute error between the swipe's points and the straight line
Median Absolute Error	The median absolute error between the swipe's points and the straight line
Coefficient of Determination	The coefficient of determination between the swipe's points and the straight line
Horizontal Acceleration	Mean acceleration of user's movement in the horizontal axis
Vertical Acceleration	Mean acceleration of user's movement in the vertical axis
Horizontal Mean Position	Mean position of user's movement in the horizontal axis
Vertical Mean Position	Mean position of user's movement in the vertical axis

Our strategy, towards developing a high accuracy model, was to develop a one-class kNN-SVM confidence-based classifier which uses the strategy "one against the universe". More precisely, we assumed that one of the users is the authorized user and all the other 974 users are the "attackers". Then, we used two primary metrics in order to evaluate the accuracy of our model, False Acceptance Rate (FAR) and False Rejection Rate (FRR). FAR is the percentage of "attackers" that was misclassified from the model as legitimate users and gain incorrectly access to the smartphone. On the other hand, FRR is the percentage of the times that the authorized user was mistakenly classified as an "attacker" from the model and the user could not access their smartphone. The mathematical equations of FAR and FRR are shown in (1) and (2) respectively.

$$FAR = \frac{attacker\ accepted\ swipes}{attacker\ total\ swipes} \quad (1)$$

$$FRR = \frac{authenticated\ user\ rejected\ swipes}{authenticated\ user\ total\ swipes} \quad (2)$$

As we can see in Table 2, there are a lot of different features that can be used for the training of our model. When it comes to the training procedure of our model, we evaluated the dataset and more precisely we tried to find which features can better distinguish the users. Also, as we will explain in System Design Section below, we also tried to find which features are similar to each user. Towards that direction, we run some clustering algorithms such as DBSCAN, Hierarchical and kMeans. The Hierarchical Complete-Linkage algorithm worked quite well and gave us a lot of information, concerning which features better distinguish each user and which features are similar to each user. It should be noted that in order to run and understand the results of Hierarchical clustering we used different sets of about 15-25 users each time we run the algorithm. This way, if the number of clusters created by the hierarchical clustering is almost the same with the number of users that are given in the algorithm, that means that the features given in the algorithm can distinguish the users quite accurately and it is a good information for the classification of our model.

III. SYSTEM DESIGN

A. *System Overview*

An overview of the system is shown in Figure 1. More precisely, as we said before we use the method "one against the universe" so we split the data of the user who we assumed as the legitimate user and all the other data are assumed as "attacker" swipes. Following, we split the user data into training data, which will be used in order to train our model, and testing data, which will be used in order to test if our model recognizes the user correctly. On the other hand, we use the "attackers" data with the purpose of testing if our model can recognize that the person using the smartphone is not an authorized user. The training-testing partition is 75%-25% respectively and we implemented that in *R Programming Language* with the *floor()* and *sample()* functions which assure a non-deterministic way of partitioning. Then, before training our model, we preprocess the training data, since the dataset derives from a real-world application and there are a lot of *Outliers* which can alter the outcome of the classification. In section B, we will discuss more about the preprocessing techniques that we used. As shown in Figure 1, we used an ensemble method for the classification, in which there are two different models, a kNN model and an SVM model, that give two different prediction results and we summarize those predictions, creating a more accurate final prediction. In Section C, we will further discuss both the parameters used for the kNN and SVM model and the calculations made for the prediction result of each model. After calculating the predictions of our models, we introduced to our system a *confidence* variable which boosted the accuracy of our system. First, we tried to classify each swipe without the confidence value and although the FAR of our model was quite satisfying, the FRR of our model, despite the fact that it was also

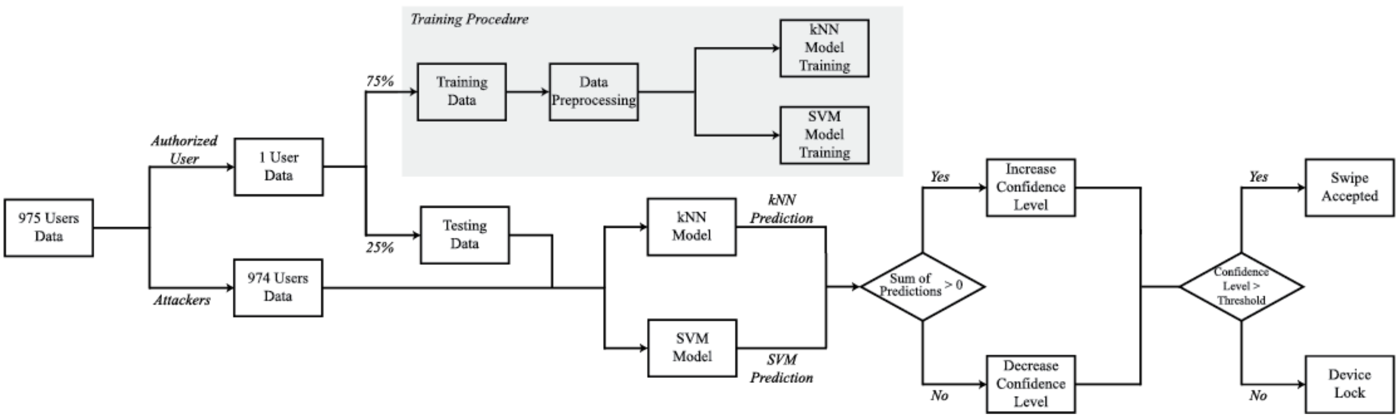


Fig. 1 System Diagram

good, we consider that it is important to be as low as possible, since the misclassification of user swipes would be irritating for the authorized user. Thus, by introducing the confidence value we achieved to bypass any misclassifications of our model, when it comes to the testing data. By adding the confidence value, we also needed to tune the *threshold* value. More precisely, if the confidence level is below the threshold value that we have defined, the device will lock.

B. Data Preprocessing

As we said before, the dataset comes from a real-world commercial application. Thus, there a lot of data swipes that should not be considered as valid data (inliers). One particular problem we discovered while watching the dataset was that there were some very quick swipes, which could be classified as taps and the some very long swipes which could be also taps. As a consequence, we classified these swipes as *Outliers.* Also, the “BrainRun” application was used in a variety of devices which had a variety of screen sizes. Thus, in order to use correctly the following features, “Horizontal Trace Length”, “Horizontal Mean Position”, “Vertical Trace Length” and “Vertical Mean Position”, we scaled all the data in the same screen size, in particular 360x640. Also, we used the *Local Outlier Factor* model, through the function *lofactor()* in *R.* This way, we eliminated some outliers which diverged a lot from the typical swiping behavior of the user. Last but not least, the most important feature used for the data preprocessing stage was the *Direction* of a swipe. Thus, we split the data from these 975 users to four different datasets, according to their direction (“Up”, “Down”, “Left”, “Right”).

C. Model Construction

During the construction of our model, a trade-off was made between accuracy and algorithmic complexity. Thus, we decided to use only two models, one kNN model with a constant number of nearest neighbors and one SVM model with constant *ν-γ* values. This way, we achieved a high efficiency and a small algorithmic complexity. More precisely, after trial & error we defined the best nearest neighbors for the training of our kNN model was 3 (neighbors = 3). Then, we introduced a *limit* value which shows if the swipe is going to be classified as user or as an “attacker”. The limit value was calculated during the training procedure and it was defined as the sum of the mean distance of all data point with its three nearest neighbors and the standard deviation of all data points of the training data multiplied by 2.5. The limit formula is shown in (3). Now, as we said before, the confidence-based modeling that we design in this work needs to know the *Certainty of Prediction (COP)* in order to result a final prediction value which is going to be added in the current confidence value. That is why we introduce the COP value which is the subtraction between the limit value and the mean value of the distance of the swipe (to be classified) with the three nearest neighbors. We can see the formula in equation (4).

$$limit = trainMeanDist + 2.5 * trainMeanDistSD \quad (3)$$

$$COP = limit - swipeMeanDist \quad (4)$$

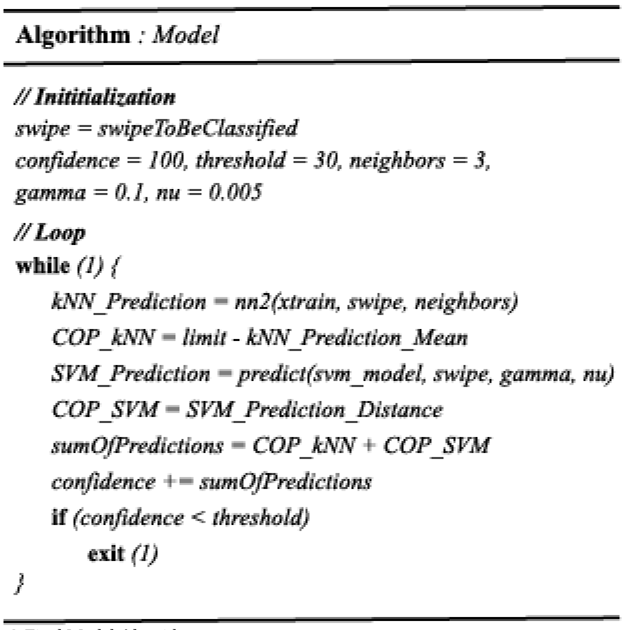
Now, for the SVM model we used the same logic and we defined, after trial & error and some trade-offs between testing error and “attackers” error, the ν-γ values as 0.005 and 0.1 respectively (ν = 0.005, γ = 0.1). Then, in order to find again the COP, we calculated the distance of the swipe (to be classified) with the SVM decision boundary. More precisely, we used the *“decision.values”* attribute that gives the *predict()* function of the SVM model in *R.* Thus, we understand that the greater the distance the greater the COP.

Then, after we calculate simultaneously, since there is no data dependency between those two classification algorithms, the COP of each model, we summarize them and we add the result to the current confidence level. If the confident level is greater that the threshold value then the smartphone continues to work uninterrupted, while if the confidence level is lower that the threshold value, the smartphone locks.

The *confidence* level was initiated at 100% since the user that unlocked the phone is assumed to be the legitimate user and the *threshold* of our system was defined at 30% after a lot of trial & error.

The features that we used for the training of both models are the ones that distinguish each user the most and they were found through Hierarchical Complete-Linkage clustering. Also, there were some features that were very similar to most of the users. Those features were being used in the training of the model only when the mean value of the training data was very different that the mean value of all users.

Following, in Figure 2 we can see the final model in pseudocode with some *R* functions, *nn2()* and *predict()* needed for the prediction of the kNN and SVM models respectively.



```
Algorithm : Model

// Inititialization
swipe = swipeToBeClassified
confidence = 100, threshold = 30, neighbors = 3,
gamma = 0.1, nu = 0.005
// Loop
while (1) {
    kNN_Prediction = nn2(xtrain, swipe, neighbors)
    COP_kNN = limit - kNN_Prediction_Mean
    SVM_Prediction = predict(svm_model, swipe, gamma, nu)
    COP_SVM = SVM_Prediction_Distance
    sumOfPredictions = COP_kNN + COP_SVM
    confidence += sumOfPredictions
    if (confidence < threshold)
        exit (1)
}
```

Fig. 2 Final Model Algorithm

IV. Evaluation

A. Evaluation Methodology

The evaluation of our model was a feedforward process between preprocessing, clustering and classification. First, we split the dataset into four smaller datasets. Each dataset for each direction. The smaller dataset consists of about 200 users each and it is designed that way so as to reassure that each user has performed a big number of swipes in each direction. This way, there are a lot of data points, both for training and for testing the model. As we said before, the method used for the evaluation of the model is the "one against the universe". At first, we evaluated each model by itself (kNN, SVM) in order to define its parameters. Also, we did not use neither the *confidence* value nor the *COP* value. For the evaluation of these models, we used the following metrics, FRR, FAR and Accuracy. As we will present in the next section, we defined the parameters of our models based on the outcomes of these metrics. After evaluating the models, we noticed that even thought the total Accuracy of the results were quite encouraging, we noticed that the FRR and the FAR of our system was significant. So, we imported the parameter of confidence. Now, in the confidence-based system, we did not use the FAR and FRR as metrics since it would be acceptable for our model to misclassify a swipe as long as the COP was not very big. Thus, the evaluation metric used for the final confidence-based model was the number of swipes that the "attacker" did before the device locks. Also, when it comes to the user's testing swipes, we took into consideration in how many swipes the device locks (because of misclassification), if it happens.

B. Evaluation Results

First, we run only the kNN model in order to establish the nearest neighbors of the final model. Towards that goal, we run 50 times (for 50 different users) the kNN model for 2-5 nearest neighbors and we evaluated the results. The results are shown in Table 3. As we can see, while the Accuracy of the model with 2 nearest neighbors is very good, the FRR of our model is significant. On the other hand, with 5 nearest neighbors, the model has very good FRR but low Accuracy. Thus, we chose our final model to have 3 nearest neighbors. So, the kNN model has a total Accuracy of about 86.3% and total FRR of about 27.4%.

TABLE 3

	UP		**DOWN**		**LEFT**		**RIGHT**	
	FRR	***ACC***	***FRR***	***ACC***	***FRR***	***ACC***	***FRR***	***ACC***
2	57.6%	95.6%	47.9%	93.7%	55.8%	92%	53.2%	93.1%
3	27.2%	89.5%	24.5%	86.3%	29.7%	84.3%	28.4%	85.2%
4	20.8%	86.2%	18.5%	81.9%	22.6%	79.4%	20.9%	80.6%
5	18.4%	83.6%	14.4%	79%	14.9%	76%	13.1%	78.1%

The same way, in order to define the parameters of the final SVM model we run for 50 different users the SVM model for different parameters. For simplicity reasons, we present the results of 3 ν-γ pairs. In Table 4 we can see the results. As we can see, while the pair 1 (0.005-0.01) has very low FRR, the total Accuracy is also low and while the pair 3 (0.005-0.2) has high total Accuracy, the FAR is also high. So, we chose the pair 2 (0.005-0.1).

TABLE 4

	UP		**DOWN**		**LEFT**		**RIGHT**	
	FRR	***ACC***	***FRR***	***ACC***	***FRR***	***ACC***	***FRR***	***ACC***
1	1.1%	72.8%	0.9%	70.4%	0.7%	69.3%	0.7%	69.9%
2	8.4%	80.2%	7.8%	79.2%	6.2%	78.1%	6.4%	79.1%
3	22.8%	86.6%	19.7%	84.3%	16.3%	82.5%	17.2%	83.6%

As we said before, although the models by themselves have a high Accuracy, the FRR and FAR of the models cannot be ignored. Thus, we inserted to our system the confidence value and the COP value. Now, instead of locking the phone instantly in case of a misclassification, when the authenticated user operates the smartphone, or the other way around, the confidence value will change according to the COP value. So, the metric used for evaluating the model is the number of swipes that the "attacker" did before the device locks. Also, we evaluate if the device locks when the authenticated user operates the device. In Table 5 we can see the results for all directions of swipes. Again, in those results are the mean value of 50 different users that was tested for each direction. From the following results for every direction, we can say that the total number of accepted swipes of our model is about 2.44 swipes.

TABLE 5

	UP	***DOWN***	***LEFT***	***RIGHT***
Number of accepted swipes	2.32	2.54	2.51	2.42

Finally, in Figure 3 we present a histogram (for one user) that in the vertical axis shows the frequency of the number of swipes that the "attacker" made before the device locked. As we can see, in most times, the "attacker" commits up to 4-5 swipes before the device locks, which is quite encouraging given the fact that we assume that in order to gain access to personal information of the authenticated user, it is needed more than 4-5 swipes.

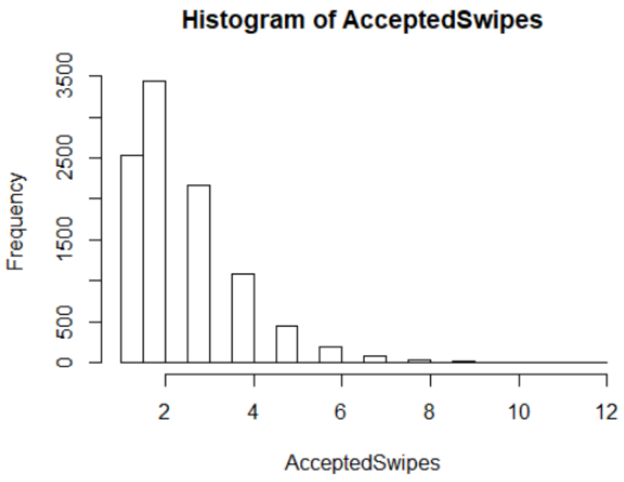


Fig. 3 Histogram of Accepted Swipes

V. Conclusions

The results from the Tables 3 and 4 show that the kNN model has a good overall Accuracy but a big FRR. On the other hand, the SVM model has low Accuracy and quite small FRR. So, by combining those two models we created a model that has both good Accuracy and small FRR. However, the FRR was still significant, so we introduced the confidence variable. The confidence-based system has the advantage that even if it occurs a misclassification of the authenticated user, the device does not lock. The only thing that happens is that the confidence level drops, according to the COP value and, possibly, in the next swipe of the user the confidence level will elevate again. The only disadvantage of the confidence-based model is that the "attacker" might be able to commit a number of swipes before the device is being locked. However, as we saw in the previous section, the number of accepted swipes is usually less than 5 swipes which is a permitted number of swipes, since an "attacker" might need more swipes in order to gain personal information. Another thing that must be noticed is that the confidence-based model cannot be compared with the previous non-confidence-based model, since the evaluation metrics that we use are different.

Another thing that we took into consideration is the complexity of our model, since as we explained before, the model is going to run every time there is a gesture. So, we only used two models instead of using more, sacrificing the Accuracy and the Accepted Swipes number of our model.

As of future work, in an attempt of improving this model, we could also use the information that the taps of the initial dataset could provide. The final step is to test our model in real world device.

References

[1] "Number of smartphones sold to end users worldwide from 2007 to 2001",online:https://www.statista.com/statistics/263437/global-martphone-sales-to-end-users-since-2007/.

[2] M. Qi, Y. Lu, J. Li, and J. Kong, "User-specific iris authentication based on feature selection," in CSSE, 2008.

[3] K. Xi, J. Hu, and F. Han, "Mobile device access control: an improved correlation based face authentication scheme and its java me application", Concurrency and Computation: Practice and Experience, 2012.

[4] "How often do you check your phone?", online: https://www.dailymail.co.uk/sciencetech/article-2449632/How-check-phone-The-average-person-does-110-times-DAY-6-seconds-evening.html.

[5] L. Yang, Y. Guo, X. Ding, J. Han, Y. Liu, C. Wang and C. Hu, "Unlocking smart phone through handwaving biometrics," IEEE Transactions, on Mobile Computing, 2015.

[6] W. Lee, and R. B. Lee, "Sensor-based implicit authentication of smartphone users," in 2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2017, pp. 309-320.

[7] T. Feng, J. Yang, Z. Yan, E. Munguia Tapia, and W. Shi "Tips: context-aware implicit user identification using touch screen in uncontrolled environments", 02 2014.

[8] M. D. Papamichail, K. C. Chatzidimitriou, T. Karanikiotis, N. C. Oikonomou, A. L. Symeonidis, and S. K. Saripalle, "Brainrun: A behavioral biometrics dataset towards continuous implicit authentication," Data, vol. 4, no. 2, p. 60, May 2019. [Online]. Available: http://dx.doi.org/10.3390/data4020060.