Keystroke Dynamics Authentication based on Principal Component Analysis and Neural Network

Dr.Shaimaa Hameed Shaker, Dr.Riyadh Jabbar Saydani, Mina Khidhir Obaid

Abstract— With the ever increasing system threats and security demands to defeat these threats, the need arises to build robust security systems. Password-based authentication systems are vulnerable to password attacks. Keystroke dynamics authentication system is considered one of the alternative modern dependable solution due to its cheapness, non-intrusiveness and user-friendliness. In this paper, five timing features which are the key duration, up-down, down-down, up-up latencies and total typing are extracted from 10 users with 20 trials for each user, and combined to be used to verify the user using MultiLayer Perceptron Neural Network (MLP NN) classifier with dimension reduction using Principal Component Analysis (PCA). Results showed enhancement in system accuracy due to PCA reduction. Not only the False Rejection Rate (FRR) and False Acceptance Rate (FAR) are dropped to 24% and 6% respectively, but also the neural network Mean Square Error (MSE) and training time are decreased with PCA deployment.

Index Terms— Feature Reduction, Keystroke Dynamics, MLP NN, PCA, Timing Features, Total Typing, User Authentication.

IJSER

1 Introduction

COMPUTER systems and network are being used in almost every aspect of our daily life. As a result, the security threats to computers and networks have also increased significantly. Besides, the conventional password-based authentication systems are vulnerable to password attacks. Password can be lost or stolen. On the other hand, keystroke dynamics, which represent the typing rhythms the user exhibits while typing on the keyboard, achieves high level of security due to its non-intrusiveness. Also, its cheap implementation is a good factor to make it user-friendly compared to fingerprint and iris scan which need additional hardware to achieve authentication, while all it is needed in keystroke dynamics is a keyboard and a code that uses the keys timings to authenticate a user. So keystroke dynamics can be combined with password-based authentication to achieve strong authentication.

Keystroke dynamics depend on the timing features extracted while the user is typing. The most common used features are represented by key duration and three key latencies: the up-down latency, the down-down latency and up-up latency. Another feature is proposed in [1] which is the total typing. These features can be used solely or combined with each other to gain better system performance.

Classification is the most important step in keystroke dynamics authentication. One of the common used classifier is the Neural Network (NN) where it is trained using the timing features samples extracted from the users' typing. The trained NN is used later on to verify the claimed user. In [2] and [3], neural network has been deployed for classification.

In fact, some researchers have used an optimization technique to enhance the system performance by reducing the dimensions using Principal Component Analysis (PCA). In [4], PCA with MLP NN is performed on Electrical Capacitance Tomography (ECT) data where results showd that using PCA as a dimension reduction improved the MLP's estimation performance and reduced the training time, while the results in [5] has showed that PCA is a linear transformation technique which less efficient for nonlinear data. The researcher in [6] has shown that dimensionality reduction using PCA provide 9.2% misclassification rate depending on the testing samples of the neural network.

The outline of this paper goes like: section 2 represents the keystroke dynamics authentication system in details. Section 3 represents the NN classifier and PCA as a feature reduction technique. Section 4 represents the proposed system. Section 5 represents the implementation of the proposed system. Section 6 shows the results of implementation. Discussion and conclusion are represented in section 7 and 8 respectively.

2 Keystroke Dynamics Authentication System

Keystroke dynamics is defined in [7] as " the process of analyzing the way users type by monitoring keyboard inputs and identifying them based on patterns in their typing rhythm " .It is based on the assumption that each typist has his own unique way in typing that recognize him from other typists.

2.1 Data Acquisition

In order to authenticate a user, the keystroke dynamics system needs data from which the typing features are extracted to build the model that will be used to verify the claimed user.

2.2 Feature Extraction

The common used timing features that can be extracted are:

- **Key duration**: also named "dwell time", the amount of time a key is pressed as shown in the following equation:

$$KeyDuration = R_i - P_i \quad (1)$$

Where:
R_i: release time of the *ith* key.
P_i: press time of the *ith* key.

- **Up-Down latency**: also named "flight time", the difference in time between the release of key and the press of the next key.

$$UpDownLatency = P_{i+1} - R_i \quad (2)$$

- **Down-Down latency**: the release times of two successive keys.

$$DownDownLatency = P_{i+1} - P_i \quad (3)$$

- **Up-Up latency**: the difference between the press times of two successive keys.

$$UpUpLatency = R_{i+1} - R_i \quad (4)$$

Another feature that can be used as an extra feature is proposed in [1]:

- **Total typing**: the total time needed to type the whole string.

$$TotalTyping = R_{i=N} - P_{i=1} \quad (5)$$

Where:
N: the numbers of the string characters.

IJSER

2.3 Keystroke Dynamics Approaches

Keystroke Dynamics algorithms can be approached in different ways. The most commonly used techniques are either based on: statistical algorithms [8], or neural networks [2], [3], [5] and [6] where the network is trained using the extracted features and saved to be used to classify the claimed user in the verification.

3 Neural Network and Principal Component Analysis

3.1 Neural Network as a Classifier

NN are massively parallel computing systems consisting of an extremely large number of simple processors called neurons with many interconnections. NN models attempt to use some organizational principles believed to be used by the human brain [9]. One of the scopes that the NN is used for is pattern recognition where an input pattern is assigned to one of many pre-specified classes.

MLP is one of the commonly used NN. It is a feed forward neural network that maps groups of input data onto a set of target outputs i.e. supervised network. Its name indicates that it consists of multiple layer: one or more of hidden layers and output layer. Each layer consists of multiple neurons.

As a learning algorithm, resilient propagation is a good candidate for fast learning and less memory consuming compared to other learning algorithms since it depends on the sign not the magnitude of the partial derivate of the error to update the weights [10].

NN has some misclassification when it is trained using correlated data because correlation causes confusion to the NN because of the redundant data and limits the generalization capability of NN training.

3.2 Principal Component Analysis as a Feature Reduction Technique

The solution to the misclassification in NN is to remove the redundant data enough to remove confusion and increase the generalization capability of NN training leading to decreasing the misclassification rate. PCA as a dimension reduction technique is responsible of transforming the correlated variables into new uncorrelated variables called "principal components"[11].

PCA is performed by computing the covariance of the normalized data. Eigenvectors and eigenvalues are computed from the covariance. Each eigenvector contains the features loadings of the new component. Eigenvalues represent the importance of the new principal components (PCs) in term of variance. They are ordered in descending way. After determining the number of the meaningful components to retain depending on the eigenvalues, the retained eigenvectors represent the transformation matrix of the original data into the new projection. It is multiplied by the original data to produce the retained PCs [12].

Different ways exist in to determine the number of meaningful components. One of them is the portion of the variance percentage one decided to retain. This portion is called "PC variance". For example, one can decide to retain 5% or 10% of PC variance [11].

The resulting PCs have the properties that: The 1st PC is uncorrelated with the other PCs and correlated with most of the original features. The 2nd PC is uncorrelated with the remaining PCs and correlated with the features that didn't show high correlation with the 1st PC and so on for the remaining PCs [11].

Sometimes, in order for PCA to perform well, normalization is done on the data before being fed to PCA. For example, z-score normalization, unity vector norm and log ratio [13].

4 The Proposed Model

4.1 System Overflow

The proposed model is illustrated in Fig. 1 where ten users are asked to register in the system. Each user i types his ten-length password j=20 times to build the database which will be needed to verify him each time he tries to have access later on. The user enters his username and password. The timing features are being extracted from the password transparently while the user is typing it. After 20 times of entries, username and password are appended to the features vectors and saved as templates in the database. PCA is performed on the features to reduce them before they are served as inputs to the NN to be trained. The trained NN is also saved in the database. The built model including the learned NN and the features templates (represented by the violet shapes) is now ready to be used to classify the user.

When the claimed user x tries to grant access to the system by typing his username and password, the same typing features are extracted from the password after checking the user validity and the password correctness by comparing them to the enrolled users' templates. These typing features are fed to the trained NN to classify the claimed user as authenticated or not.

4.2 Data Collection

Ten users were asked to type their usernames and their own passwords with the condition that the password should be of length 10 for over four sessions of two weeks-period each, five times for each session. So the total collected samples are 200 samples. Samples are collected using Dell Inspiron keyboard and the users were sitting on the same chair under the same lighting conditions.

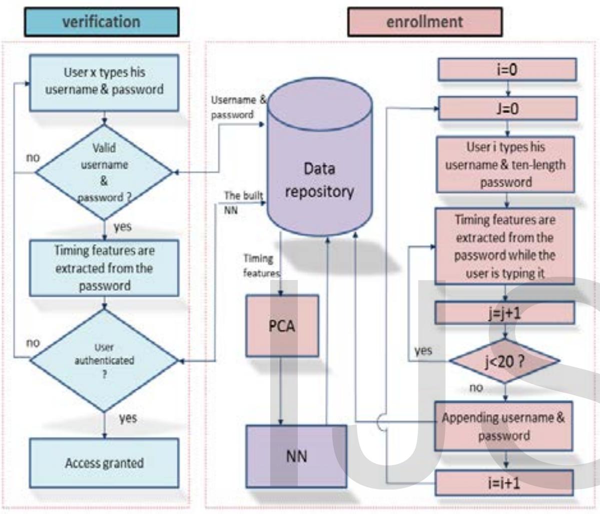

verification
enrollment
User x types his username & password
Valid username & password ?
no
yes
Timing features are extracted from the password
User authenticated ?
no
yes
Access granted
Username & password
Data repository
The built NN
Timing features
PCA
NN
i=0
J=0
User i types his username & ten-length password
Timing features are extracted from the password while the user is typing it
j=j+1
yes
j<20 ?
no
Appending username & password
i=i+1

Fig. 1. The proposed keystroke dynamics authentication system overflow.

4.3 Feature Extraction

While users are typing their own password, the five timing features explained in section 2.2 were extracted. For each trial, these features are combined where the key durations (KD) of the 10 letters are followed by the 9 Up-Down (UD) latencies, followed by the 9 Down-Down (DD) latencies, followed by the 9 Up-Up (UU) latencies and followed by the Total Typing (TT) time constructing a one features vector of length 38 as shown in Fig. 2. 200 samples of features vectors are saved in a database.

4.4 Feature Reduction using PCA

In order to enhance the system performance and reduce the learning time of NN, PCA is used as a preprocessing step to reduce the features before learning the NN with these features vectors. Due to the inconsistency in the features vectors i.e. the total typing time is much more than the other features as Fig. 2 shows, z-score normalization is performed on the features vectors before using PCA.

PCA is performed as in section 3.2. The input is the 200 samples of the features vectors which are of length 38. The features are reduced so that the minimum PC variance percentage is 5% which will retain enough PCs to achieve more than 80% of the cumulative percentage of the total variance. The result is 200 samples of 5 components instead of 38 which represents the inputs to the NN.

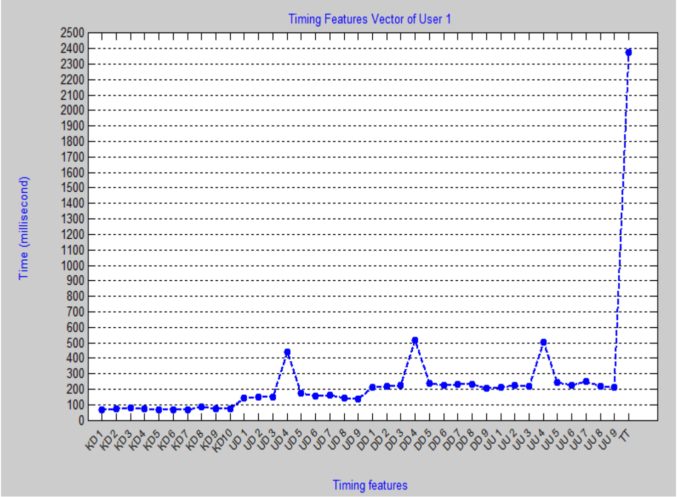


Fig. 2. The timing features vector of user #1.

4.5 Model Building using MultiLayer Perceptron Neural Network

MLP NN is used to build the model which will be later used in user verification. It is implemented as a pattern recognition neural network using matlab script. The proposed NN is a two-layered feed-forward network with 70 sigmoid hidden neurons and 10 sigmoid output neurons as shown in Fig. 3.

After the features have been normalized and reduced, they are served as inputs to the NN with the desired targets. The 200 samples are divided as 80% for training the network, 10% for validation which is used to stop the network and 10% for testing the trained network. *Pattern Recognition Neural Network* randomly generates the initial weights and biases. So all the network parameters (inputs, targets, initial weights and biases) are now set and the network is ready to be trained. Fig. 4 shows the detailed architecture of the network.

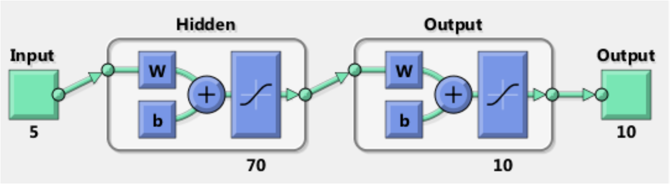


Fig. 3. The general architecture of the NN used.

The proposed NN is trained using the resilient propagation (RPROP) technique for fast learning. The trained NN is saved to be used for verification.

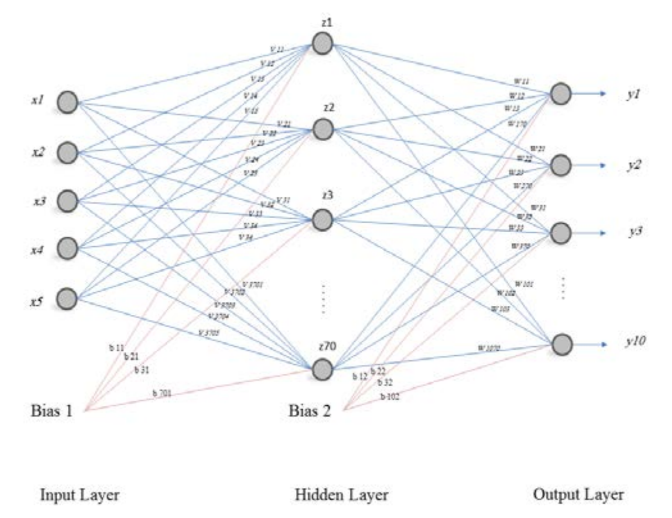


Fig. 4. The detailed architecture of the NN used.

4.6 User Verification

User is verified using the previously-enrolled samples and the trained network. The same timing features are extracted and combined in one features vector of length 38 as explained in section 4.3, and multiplied by the transformation matrix obtained from PCA to reduce the vector length to the same size of the new components. The reduced vector is also multiplied by the standard deviation to remove its effect.

Classification is done by achieving two checking: validity checking and authenticity checking respectively.

- **Validity checking:**

Usernames and passwords are retrieved from the database in order to check the validity of the claimed user by comparing the newly entered username and password with the ones retrieved from the database. If the username is valid and the password is correct then the database position of this user is obtained to be used in the authenticity checking.

- **Authenticity checking:**

After checking that the claimed user is a valid user and his password is correct, his reduced features vector is tested using the trained NN. The output is a vector of users' scores ranging between 0 and 1. If the score of the user that is in the position obtained from validity checking step is equal or greater than a predefined error threshold then the user is authenticated and grants access to the system, and if not, he is not authenticated and can try again.

The error threshold is chosen to be between 0 and 1. It defines the level of the system security. The higher threshold value the more security gained.

5 Implementation of the Proposed Model

The proposed model is implemented using Windows Forms Application on Visual Studio 2012 environment, except for PCA and NN which are constructed using MATLAB 2013. The main interface is the windows forms application while MATLAB acts as a COM server.

Fig. 5 shows the interface of the proposed model where in the enrollment, the user registers by entering username and 10-length password and press the *Enroll* button which will open a new window for extracting the features from the password as shown in Fig. 6. By pressing the *Save* button, the features vectors are saved in MS Access 2013 database and a status indicates the success of the process is shown in the status field in Fig. 5.

After the features are extracted from the users, PCA is performed by pressing the *Perform PCA* button which will execute the matlab code of PCA. The result is the reduced features and saved in a mat file to be the input to the NN. By pressing the *Learn NN* button, the network is trained using *Pattern Recognition Neural Network* matlab tool, and saved also as a mat file to be used to verify the user.

In the verification, the claimed user enters his username and password and presses the *Check* button in Fig. 5 which will extract the features vector and do the validity by using a function to get the usernames and passwords from the database and comparing the username and password strings of the claimed user with the retrieved ones. After that, authenticity checking is done. If the user is authenticated, a message box saying "User is authenticated" is displayed, if not, a message box saying "User is not authenticated" is displayed.

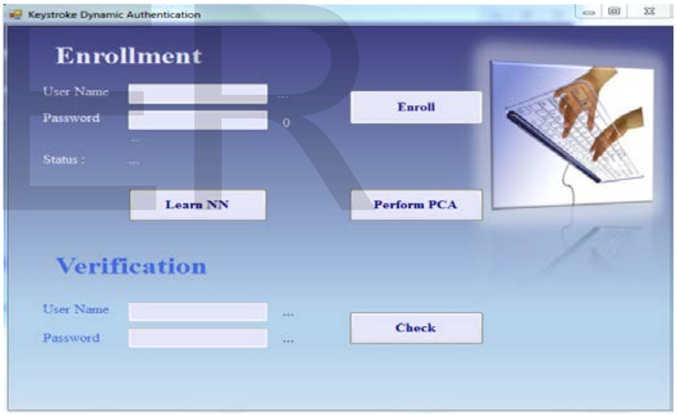
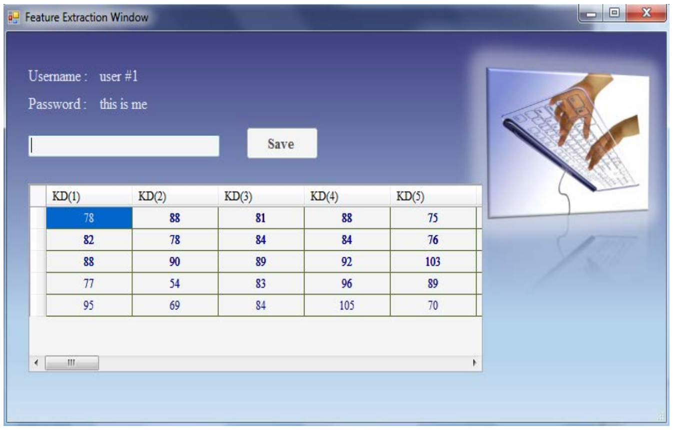


Fig. 5. The interface window of the proposed model.



Username : user #1
Password : this is me

KD(1)	KD(2)	KD(3)	KD(4)	KD(5)
78	88	81	88	75
82	78	84	84	76
88	90	89	92	103
77	54	83	96	89
95	69	84	105	70

Fig. 6. The feature extraction window.

IJSER

6 Results

6.1 PCA Results

Fig. 7 shows that only the first 5 components have PC variances above 5%. So the result of PCA is 5 components of 200 samples each, resulting from multiplying the normalized data by the transformation matrix which is of size 38 x 5. The 38 is the number of the original features and 5 is the number of the new reduced components.

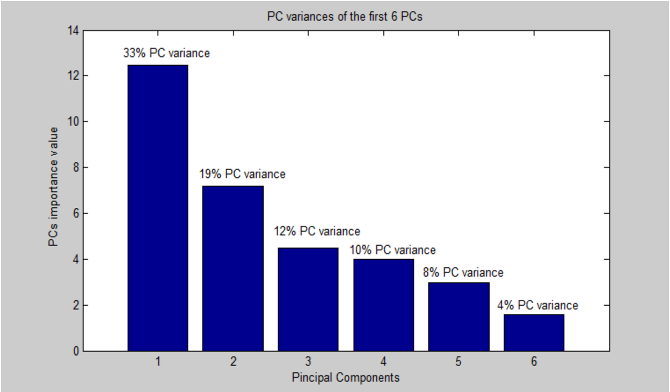


Fig. 7. The PC variances of the first 6 components.

As mentioned in section 3.2, the resulting PCs are uncorrelated with each other. Table 1 proves this by showing the correlation between each PC with the other PCs equal to zero except with itself.

IJSER

TABLE 1
Correlation between PCs

	PC 1	PC 2	PC 3	PC 4	PC 5
PC 1	1	-	-	-	-
PC 2	0	1	-	-	-
PC 3	0	0	1	-	-
PC 4	0	0	0	1	-
PC 5	0	0	0	0	1

Table 2 shows that the 1st PC has almost high correlation with most of the features i.e.values in red with more than 0.5 regardless of the sign. The 2nd PC has almost high correlation with the variables that did not show high correlation with the 1st PC and so on.

6.2 NN Results

This section shows the results of implementing the trained NN on the testing samples.

- NN confusion between outputs and targets

The total misclassification has dropped from 2% to 1% with PCA deployment. Fig. 8 and Fig. 9 show the confusion between the output and targets of NN without and with PCA deployment respectively.

TABLE 2
Correlation of PCs with the Features

PC/ var	PC 1	PC 2	PC 3	PC 4	PC 5
var 1	0.659618	0.327821	-0.08898	0.218996	0.061299
var 2	0.315285	0.655911	-0.29298	-0.25587	-0.19355
var 3	0.536419	0.364602	-0.20197	-0.12041	-0.33721
var 4	0.65993	0.519447	-0.10313	-0.12907	0.276922
var 5	0.533959	0.475288	-0.23049	-0.09979	-0.1121
var 6	0.541563	0.391825	-0.39648	0.355333	0.145313
var 7	0.4261	0.388117	-0.28396	0.04245	-0.24347
var 8	0.528121	0.379884	-0.51175	-0.04507	0.212873
var 9	0.391596	0.564312	-0.20744	-0.04823	-0.44528
var 10	0.562081	0.306379	-0.04866	-0.42153	-0.03315
var 11	0.483382	-0.43452	0.412744	-0.48817	0.205048
var 12	-0.41649	0.78141	0.337453	-0.14542	0.038207
var 13	0.681642	0.230571	-0.00492	0.588391	0.140002
var 14	-0.38899	0.156328	-0.474	-0.6104	-0.25981
var 15	0.649976	-0.23555	0.48568	-0.36229	0.008289
var 16	-0.75281	0.241193	0.297014	-0.03001	0.368316
var 17	0.639667	-0.22057	0.383248	0.134044	-0.54038
var 18	0.689519	0.316127	-0.05693	-0.15925	0.450202
var 19	-0.38436	0.620304	0.583898	0.142436	-0.23815
var 20	0.573226	-0.37809	0.392279	-0.44702	0.210601
var 21	-0.35978	0.826803	0.286935	-0.17015	0.012971
var 22	0.70503	0.252091	-0.01981	0.565228	0.111565
var 23	-0.3227	0.211892	-0.48827	-0.62839	-0.2327
var 24	0.704937	-0.15082	0.428245	-0.36197	-0.00962
var 25	-0.70005	0.310435	0.248345	0.022943	0.405022
var 26	0.668332	-0.16663	0.336093	0.134463	-0.5507
var 27	0.702781	0.338483	-0.11334	-0.15322	0.443271
var 28	-0.34026	0.661281	0.553228	0.135176	-0.27532
var 29	0.535322	-0.35101	0.379165	-0.53196	0.181712
var 30	-0.35717	0.818464	0.314562	-0.15801	0.001602
var 31	0.709245	0.270833	-0.01539	0.539271	0.160431
var 32	-0.34139	0.197624	-0.49318	-0.61784	-0.26908
var 33	0.740308	-0.1647	0.412056	-0.29688	0.033821
var 34	-0.72661	0.291013	0.273825	-0.02614	0.351617
var 35	0.746543	-0.1479	0.285886	0.126089	-0.50223
var 36	0.701788	0.34772	-0.06977	-0.15923	0.410904
var 37	-0.32416	0.660312	0.583462	0.096028	-0.24397
var 38	0.536549	0.643277	0.320705	-0.2792	0.072762

Var = features; PC = Principal Component

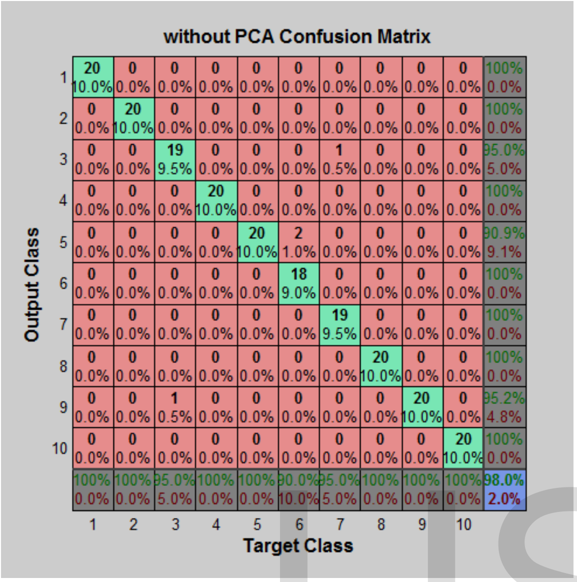
Positive value indicates positive linear relationship: as that variable increases in its values, that PC also increases in its values via an exact linear rule.
Negative value indicates negative linear relationship: as that variable increases in its values, that PC also decreases in its values via an exact linear rule.

- MSE performance

Fig. 10 shows best MSE performance is 0.012 at epoch 33 without PCA deployment, while Fig. 11 shows best MSE performance is 0.0005 at epoch 22 with PCA deployment.

- NN training time

The time needed to train the NN has dropped from 0.359 sec to 0.224 sec with PCA deployment. Fig. 12 shows the training time of each epoch without and with PCA deployment.

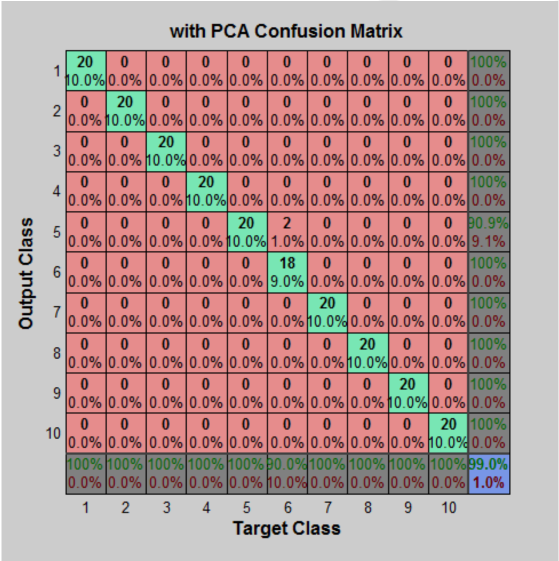


without PCA Confusion Matrix

Output Class / Target Class

	1	2	3	4	5	6	7	8	9	10	
1	20 10.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
2	0 0.0%	20 10.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
3	0 0.0%	0 0.0%	19 9.5%	0 0.0%	0 0.0%	0 0.0%	1 0.5%	0 0.0%	0 0.0%	0 0.0%	95.0% 5.0%
4	0 0.0%	0 0.0%	0 0.0%	20 10.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
5	0 0.0%	0 0.0%	0 0.0%	0 0.0%	20 10.0%	2 1.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	90.9% 9.1%
6	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	18 9.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
7	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	19 9.5%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
8	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	20 10.0%	0 0.0%	0 0.0%	100% 0.0%
9	0 0.0%	0 0.0%	1 0.5%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	20 10.0%	0 0.0%	95.2% 4.8%
10	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	20 10.0%	100% 0.0%
	100% 0.0%	100% 0.0%	95.0% 5.0%	100% 0.0%	100% 0.0%	90.0% 10.0%	95.0% 5.0%	100% 0.0%	100% 0.0%	100% 0.0%	98.0% 2.0%

Fig. 8. Confusion between outputs and targets without PCA



with PCA Confusion Matrix

Output Class / Target Class

	1	2	3	4	5	6	7	8	9	10	
1	20 10.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
2	0 0.0%	20 10.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
3	0 0.0%	0 0.0%	20 10.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
4	0 0.0%	0 0.0%	0 0.0%	20 10.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
5	0 0.0%	0 0.0%	0 0.0%	0 0.0%	20 10.0%	2 1.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	90.9% 9.1%
6	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	18 9.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
7	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	20 10.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
8	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	20 10.0%	0 0.0%	0 0.0%	100% 0.0%
9	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	20 10.0%	0 0.0%	100% 0.0%
10	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	20 10.0%	100% 0.0%
	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	90.0% 10.0%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	99.0% 1.0%

Fig. 9. Confusion between outputs and targets without PCA

6.3 Verification Results

This section shows implementing the trained NN on the extracted features in the verification step.

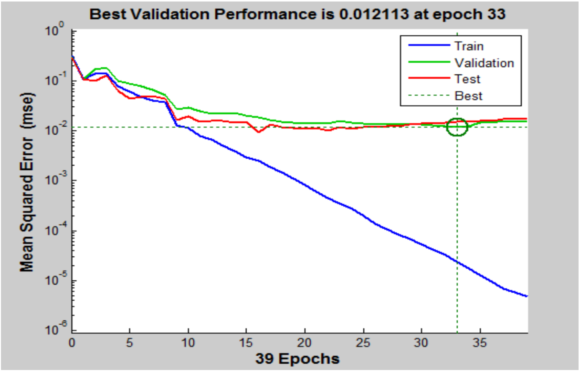


Fig. 10. MSE validation performance without PCA

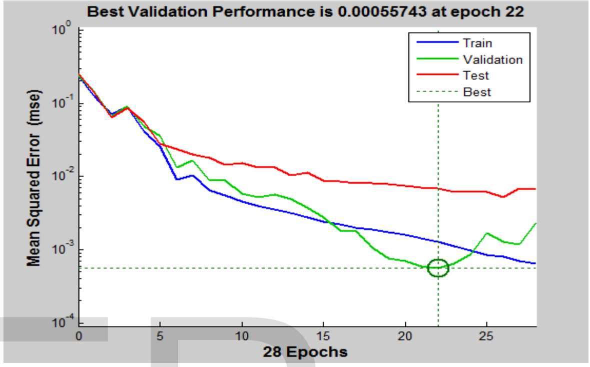


Fig. 11. MSE validation performance with PCA

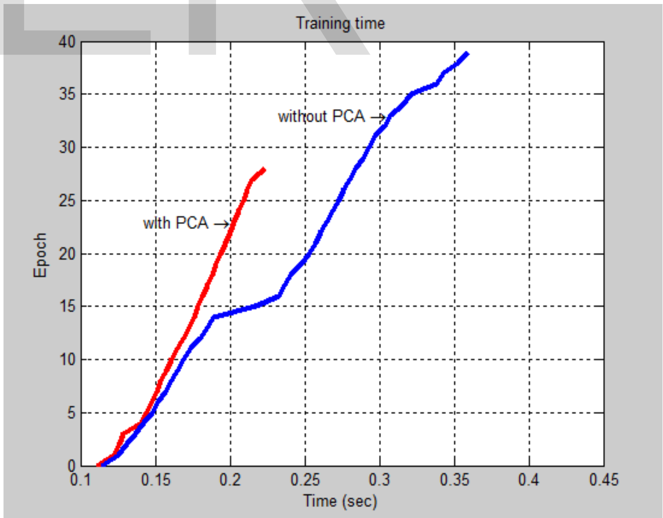


Fig. 12. NN training time with and without PCA

The proposed system is evaluated using two rates:

- FAR (False Acceptance Rate): the rate of incorrectly authenticate an imposter.
- FRR (False Rejection Rate): the rate of incorrectly reject an authentic user.
- EER (Equal Error Rate): the value where FAR and FRR are equal.

At high security level where the error threshold is 0.9, FRR has dropped from 53% to 36% with PCA deployment, while FAR is kept the same 0%. With the error threshold used in the proposed system which is 0.5, FRR has dropped from 34% to 24%, while FAR has dropped from 10% to 6% with PCA deployment.

Fig. 13 and Fig. 14 show FAR, FRR and EER with different error thresholds without and with PCA deployment respectively.

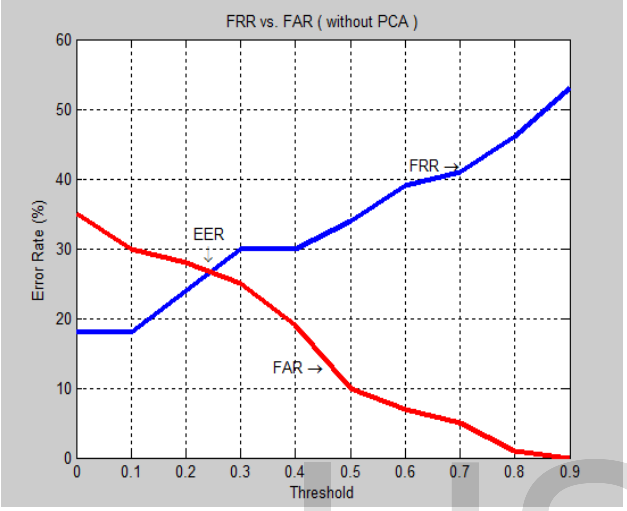


Fig. 13. FAR, FRR and EER for different error thresholds without PCA

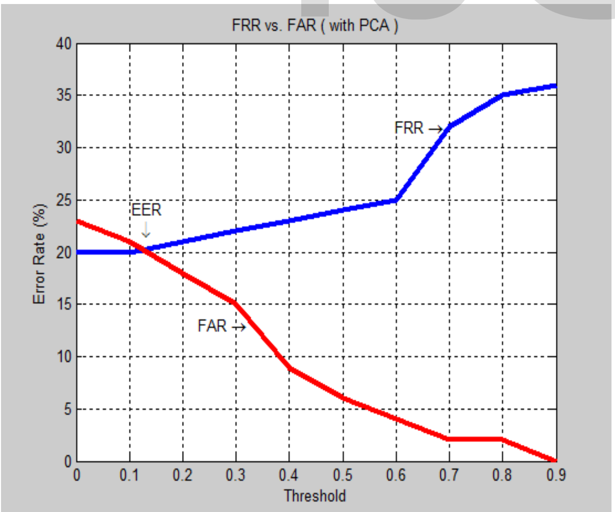


Fig. 14. FAR, FRR and EER for different error thresholds with PCA

6.4 Results Summary

Table 3 summarizes the results obtained from the proposed system compared with the results obtained without reducing the features using PCA. Results includes the misclassification rate of the trained NN, the best MSE validation performance, the time needed to train the NN, FAR and FRR at error threshold of 0.5.

TABLE 3
RESULTS SUMMARY

	Misclassi-fi-cation rate	MSE/ epoch	NN Training time (sec)	FAR	FRR
Without PCA	2%	0.012/33	0.359	10%	34%
With PCA	1%	0.0005/22	0.224	6%	24%

7 CONCLUSION

In the proposed model, five time features have been extracted and combined to be used in user verification. These features are the key duration, the three latencies (up-down, down-down, up-up), and the not-widely-used feature till now which is total typing.

In term of security, the use of total typing as an extra feature appended to the end of the features vector increases the system security. Total typing adds inconsistency to the features vector due to its far value from the other features' values. So features vectors are normalized to remove the inconsistency. Inconsistency and normalization add complexity to the system resulting in better security.

In term of accuracy, as a keystroke dynamics authentication system, the proposed system authenticate a user with 6% FAR and 24% FRR. From table 3, one can notice that these two rates represent reduced values compared with system doesn't use PCA reduced features. The reason for the proposed system to achieve these reduced rate is that it used PCA to reduce the timing features. Reducing the features removes the redundant data so that the resulting reduced components are uncorrelated with each other. Uncorrelated data increases the generalization capability of the NN learning resulting in better classification and less training time because the NN diverges faster to the desired output as shown in MSE/epoch value in table 3.

REFERENCES

[1] Romain Giot, Mohamad El-Abed and Christophe Rosenberger, "Biometrics: Chapter 8: Keystroke Dynamics Overview", pp. 157–182, Jun. 2011.

[2] David Reby, Sovan Lek, Ioannis Dimopoulos, Jean Joachim, Jacques Lauga, "Artificial neural networks as a classification method in the behavioural sciences", ELSEVIER, Behavioural Processes, vol. 40, pp. 35–43, 1997.

[3] Preet Inder Singh, Gour Sundar Mitra Thakur, "Enhanced Password Based Security System Based on User Behavior using Neural Networks", *International Journal of Information Engineering and Electronic Business(IJIEEB)*, vol. 4, no. 2, pp. 29-35, April 2012.

[4] Junita Mohamad-Saleh, Brian S. Hoyle, "Improved Neural Network

IJSER

Performance Using Principal Component Analysis on Matlab", *International Journal of The Computer, the Internet and Management,* vol.16, no.2, pp. 1-8, May-August, 2008.

[5] N. Harun, S. S. Dlay and W. L. Woo, "Performance of Keystroke Biometrics Authentication System Using Multilayer Perceptron Neural Network (MLP NN)", *Proc. IEEE Symp. Communication Systems Networks and Digital Signal Processing (CSNDSP),* pp. 711 - 714, Jul. 2007.

[6] Sucheta Chauhan, Prema K.V., "Effect of Dimensionality Reduction on Performance in Artificial Neural Network for User Authentication", *Proc. IEEE 3rd International. Advance Computing Conference (IACC),* pp. 788 - 793, Feb. 2013.

[7] Patrick Elftmann, "Secure Alternatives to Password-based Authentication Mechanisms", Diploma thesis, Laboratory for Dependable Distributed Systems, RWTH Aachen University, Aachen, Germany, October 2006.

[8] Yu Zhong, Yunbin Deng, Anil K. Jain, "Keystroke Dynamics for User Authentication", *Proc. IEEE Computer Society Conference. Computer Vision and Pattern Recognition Workshops (CVPRW),* pp. 117 - 123, Jun. 2012.

[9] Anil K. Jain, Jianchang Mao,"Artificial Neural Network: A Tutorial", *IEEE Computer*, vol. 29, no. 3, Mar 1996.

[10] Martin Riedmiller, "Advanced Supervised Learning in Multi-layer Perceptrons - From Backpropagation to Adaptive Learning Algorithms", ELSEVIER, Computer Standards & Interfaces, vol. 16, no. 3, pp. 265–278, Jul 1994.

[11] Norm O'Rourke, Larry Hatcher, "A Step-by-Step Approach to Using SAS for Factor Analysis and Structural Equation Modeling *Second Edition,* Chapter 1 PRINCIPAL COMPONENT ANALYSIS", 2013.

[12] Lindsay I Smith, "A tutorial on Principal Components Analysis", https://www.google.iq/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CCEQFjAA&url=https%3A%2F%2Fwww.ce.yildiz.edu.tr%2Fpersonal%2Fsongul%2Ffile%2F1097%2Fprincipal_components.pdf&ei=Ip-cU6DiBtH30gXJ5YCAAg&usg=AFQjCNFiLc0xvwIoZ6Iz3d3mTTkgvCJGsg&sig2=7Jt8TtHR0kz5mnBXbAUPtg, 2002.

[13] Matthias Scholz, Joachim Selbig, "Visualization and Analysis of Molecular Data", https://www.google.iq/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&cad=rja&uact=8&ved=0CCsQFjAB&url=http%3A%2F%2Fwww.researchgate.net%2Fpublication%2F6759081_Visualization_and_analysis_of_molecular_data%2Ffile%2F8f0e9966f3839de13cff25f0e1c1e44c.pdf&ei=JJycU5kFuiw0QWo3IDoCw&usg=AFQjCNFjF3UFnj12Zpwe6OsTuHonOfTbQQ&sig2=oQ512_fut5pwlPb6D1QRzw, 2007.

IJSER