Ministry of Higher Education

and Scientific Research

University of Diyala- College of Science

Department of Computer Science

Digital Signature System Based On 3D Facial Recognition In Real Time

A Thesis

Submitted to the Department of Computer Science\ College of Science\ University of Diyala in a Partial Fulfilment of the Requirements for the Degree of Master in Computer Science

By

ASRAA SAFAA AHMAD

Supervised by

Assist. Prof. Dr .Taha Mohammad Hasan

Assist. Prof. Dr. Firas Abdul hamed

2019 **1440**

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

اقْرَأْ بِاسْمِ رَبِّكَ الَّذِي خَلَقَ ﴿١﴾ خَلَقَ الْإِنسَانَ مِنْ عَلَقٍ ﴿٢﴾ اقْرَأْ وَرَبُّكَ الْأَكْرَمُ ﴿٣﴾ الَّذِي عَلَّمَ بِالْقَلَمِ ﴿٤﴾ عَلَّمَ الْإِنسَانَ مَا لَمْ يَعْلَمْ ﴿٥﴾

صدق الله العظيم

سورة العلق الآية من (1-5)

Dedication

To all who have been with me in every step in this way

To my family

To My Mother, My Father

To My Brothers and sisters

To my great country …Iraq

Asraa

2019

Acknowledgment

First, I thank Allah who helped me to complete this work and pass all the difficulties.I would like to express my sincere gratitude to my supervisors Dr .Taha Mohammad Hasan and Dr .Firas Abdul hamed on their advice and guidance that contributed to the completion of the thesis.

Asraa

2019

No	Contents	*Page No.*
	CHAPTER ONE ***GENERAL***	**1-7**
1.1	Introduction	1
1.2	Related Works	3
1.3	Problem Statement	5
1.4	Aim of The Thesis	6
1.5	Thesis Layout	7
	CHAPTER TWO ***THEORETICAL BACKGROUND***	8-32
2.1	Introduction	8
2.2	Digital Signature	8
2.2.1	Cryptography	9
2.2.2	Types of Cryptography	9
2.2.3	Factors of Digital Signature	10
2.3	Digital Signature Algorithm (DSA)	11
2.4	Cryptographic Hash Function	11
2.4.1	SHA-2 Hash Function	13
2.5	Biometric	16
2.5.1	Types of Biometrics	16
2.5.2	Biometric Recognition	16
2.6	Face Recognition	18
2.7	Real-Time Face Recognition System	18
2.7.1	Face Detection and Cropping	18
2.7.2	Image Resizing	20
2.7.3	Standardization (Normalization)	21
2.7.4	Face Feature Extraction	22
2.7.5	Face Matching	24
2.8	semi-supervised learning (SSL)	27
2.9	Authentication Performance Measures	27
2.10	Popular Authentication-Related Threats	30

2.10.1	Brute-Force Attack	30
2.10.2	Dictionary Attack A dictionary attack	30
2.10.3	Phishing Attacks	31
2.10.4	Shoulder-Surfing Attack	31
2.10.5	Guessing attack	32
2.11	3D Shapes	32
	CHAPTER THREE **THE PROPOSED SYSTEM**	**33-54**
3.1	Introduction	33
3.2	Proposed System	33
3.2.1	Registering Users	35
3.2.2.	Digital Signature Generation	47
3.2.3.	Signature Verification	52
3.2.1	Registering Users	35
	CHAPTER FOUR **Experimental Results and Performance Evaluation**	**55-75**
4.1	Introduction	54
4.2	Robustness and Uniqueness Performance Measure	55
4.3	Results of Face Detection and Recognition	56
4.4	Results of Liveness Detection	60
4.5	Results of Random Seeds Generation	63
4.6	Results of Keys Generation and Validation	70
4.7	Attacking the Generated Keys	74
	CHAPTER FIVE **CONCLUSION AND FUTURE WORK**	**76-77**
5.1	Conclusions	76
5.2	Suggestions for Future Works	77
	References	**78-84**

List of Figures

Figure No.	Caption	Page No.
2.1	Examples of different biometric traits: (a) physiological traits, and (b) behavioral traits	17
2.2	Bilinear Interpolation	20
2.3	A convolutional neural network with three-dimensional input	23
2.4	A sample convolutional neural network	24
2.5	Facenet Block Diagram	25
2.6	Triplet Loss Training	26
2.7	The relationship between the FRR, FAR, and EER	29
3.1	Overview of the proposed digital signature system	34
3.2	Liveness detection neural network	38
3.3	Random seed generation neural network	40
3.4	Optimal distribution of the values to be assigned to individuals in a sample training dataset	43
3.5	The implemented neural network for users authentication	50
4.1	ROC curve of the face authentication using the Euclidean distance	59
4.2	Spoofing attack videos collection setup	60
4.3	Sample frames of fake and real videos, Left: Real video frame, Right: Fake video frame	61
4.4	Distribution of the prediction's centroid, normalized centroids and optimal distribution; Left: 1st iteration; Center: 500th iteration; Right: 1000th iteration	64
4.5	Histogram of the number of batches required to produce the users' private keys	73

List of Tables

Table No.	*Caption*	*Page No.*
2.1	The confusion Matrix used to Calculate Performance Measures of an Authentication Technique	27
4.1	Performance measures of the MTCNN face detection using the collected real-time dataset	56
4.2	Confusion matrix of the predictions of the face authentication neural network	59
4.3	Summary of the performance measures of the proposed face authentication neural network	60
4.4	Confusion matrix of the predictions of the liveness detection neural network	61
4.5	Summary of the performance measures of the proposed liveness detection neural network	61
4.6	Minimum accuracy for real and fake videos liveness detection	62
4.7	Performance of the random seed generation neural network using the test split of the FERET dataset	65
4.8	Performance of the random seed generation neural network using the videos collected from the volunteers	68
4.9	The number of batches and time required to generate the private and public keys during registration and signing phases	72
4.10	attack resistance stages in the proposed method	74

List of Abbreviations

Abbreviations	Meaning
CNN	Convolutional Neural Networks
DS	Digital Signature
DSA	Digital Signature Algorithm
DSS	Digital Signature Standard
EER	Equal Error Rate
FAR	False Acceptance Rate
FRR	False Rejection Rate
MTCNN	Multi-Task Cascade Networking
ReLU	Receiver Operating Characteristic
ROC	Rectified Linear Unit
SHA	Secure Hash Algorithm
SSL	semi-supervised learning
CNN	Convolutional Neural Networks
DS	Digital Signature
DSA	Digital Signature Algorithm

Abstract

Digital Signature (DS) has gained significant attention in recent years, according to the rapidly growing use of digital information to access several services. The responsibility of a DS system relies mainly on the security of the users' secret keys, as these keys are used to prove the authenticity of the received data and their integrity. Moreover, as long as the sender's secret key is not known to any other individual, such as an attacker, the sender cannot repudiate that the document is originated from them. Thus, a novel method is proposed in this thesis to extract the users' secret keys directly from their facial features, without storing them in the database. Hence, signing a document requires the physical presence of the sender and compromising the database does not allow the attacker to produce false signatures. Hence, the responsibility of a DS system of the proposed system is significantly higher than any other system existing in the literature.

The proposed system follows the Digital Signature Standard (DSS) using the SHA-256 hashing function to produce a unique hash for each digital document being signed and the Digital Signature Algorithm (DSA) to encrypt and validate the hashed of the sent and received documents. The secret key of a user in DSA requires a prime number of certain characteristics. The proposed system uses a neural network to produce a random seed that is used to generate the required prime number. This neural network is trained using a semi-supervised approach, that enables the production of robust and unique random seeds. This approach requires the optimal distribution of the required values for the users. Then, each user is assigned with a certain optimal value based on the predictions of the neural network. Hence, the assigned values still maintain the features detected by the neural network but also enforces the characteristics defined by the optimal values.

As the values produced by the random seeds generation neural network are not guaranteed to be unique and robust per each user, the generated secret keys are validated using the user's public keys stored in the database. Additionally, to improve the security of the proposed system and immunize it against certain attacks, such as spoofing, a neural network is trained to detect liveness in the videos collected for the users in real-time. Moreover, to secure the proposed method against shoulder surfing and phishing attacks, the users are required to authenticate using live videos captured for their faces after authenticating using their usernames and passwords.

A batch of 30 face images is collected in real-time from the user and used in every stage on the proposed method. The experimental results conducted using real-time videos collected from 33 volunteers show that the registration of the users of the proposed system requires an average of 1.082 second per user and 4.285 for signing documents. The values produced by the random seeds generation neural network have shown 19.72% robustness and 100% uniqueness, using the same videos collected from the volunteers. Moreover, the liveness detection neural network has shown 99.96% F1-score, whereas the user authentication method achieved 93.92%.

Chapter One

Chapter One

General

1.1 Introduction

With the rapid development and ease of access to the internet, digital services are being widely used to access different types of services [1, 2]. Some of these services, such as online banking, e-commerce, and e-governments, require the exchange of important and sensitive information. Moreover, the number of digital documents being exchanged over computer networks, including the internet, is increasing exponentially [3, 4].

However, the techniques being used to manipulate the contents of information being exchanged over computer networks are also developing. Hence, concerns about the integrity and security of information exchanged by users are growing rapidly, which brings significant attention to the techniques that can ensure such integrity [5].

Cryptography techniques can be used to cipher the information being communicated, where gaining access to the communicated information is of no value as it cannot be deciphered. Hence, the data being communicated is kept secret and protected from manipulation, as the manipulated data are also required to be ciphered using keys that are normally kept secret [6].

Cryptography techniques are categorized into two main categories, symmetric and asymmetric. In symmetric cryptography, a pre-shared key is used for both encryption and decryption procedures, so that the same key can be used by both partners to secure their communications.

In asymmetric cryptography, each user has a secret key and one or more public keys, where the key used to encrypt the information cannot be used to decrypt it [7].

Normally, when the information being communicated is being encrypted the receivers, public key is used to encrypt it, so that no other user can retrieve the original information as the secret key is only known to the receiver [8].

Moreover, some of the cryptography techniques can produce multiple public keys for a single user, based on a single secret key, to improve the security of the communication. The use of several public keys denies the possibility of predicting the actual data by encrypting them, using the same public key, so that even when the same piece of information is being encrypted by multiple senders, the resulting encrypted message is not similar to each other's [9, 10].

One of the widely used cryptography techniques that have been used in different applications is Digital Signature (DS). Such techniques do not encrypt the information being communicated but can be used to prove the integrity and authenticity of the received information. Instead, DS techniques compute a unique string, known as hash, that represents the contents of the message being communicated and encrypt it, i.e. the hash, using the senders secret key.

Hence, the receiver can verify the integrity of the received information, by locally computing the same hash, and comparing it to the encrypted hash received with the message, using the senders public key. Thus, the public key in DS is actually used for decryption [11, 12].

Additionally, the use of biometric information to authenticate users has been employed in many modern applications, according to their high robustness, availability, and immunity against several attacks [13].

A biometric template can represent physical features extracted from a certain body part or behavioral features collected from the user during the execution of a certain task. The use of physical templates, such as the fingerprint, face and iris, has shown significantly higher robustness,

compared to the user of behavioral features. Moreover, according to the high availability and ease to acquire facial features, face recognition is being widely used to recognize and authenticate users [14, 15].

Traditional authentication techniques suffer from vulnerability toward simple attacks, such as guessing and shoulder surfing, especially with the tendency of users to use easy-to-remember secrets, such as passwords and patters. Thus, and according to the importance of the users' secret keys, biometric authentication has been widely used to restrict users' access to those keys, so that only the legitimate user has the ability to retrieve their key [16, 17].

However, storing these keys in any format can still impose a threat toward gaining access to them, as the attacking techniques to breach the security of the systems being used to store such data, such as database servers, are developing rapidly. Gaining access to the users' secret keys allows the attackers to sign any document, which dramatically reduces the liability of the DS system [18].

1.2 Related Work

Several related methods to the proposed work in this study have been proposed in the literature.

1- ***Y. Hussain and I. Saleem*** 2015 [21] produce a new real-time surveillance system in video which makes use of face characteristics of the user for correct identification. The face is first detected using Viola-Jones algorithm, then a hybrid algorithm were used to extract the features and determine the faces linear discriminate analysis and local binary patterns.

2- ***M Shujah Islam et al*** 2016 [23] propose a real-time face recognition system based on computer vision techniques. The system extracts face images based on the use of viola jones

cascading filters for faces detection, which are then cropped out of the image before the features extraction, using the SURF method. The extracted features are matched using the M-estimator Sample Consensus (MSAC) in order to authenticate users. The method has been able to achieve 95.9% recognition accuracy, which shows the importance of using face detection prior to matching stage, to eliminate any additional features that may exist in the image.

3- ***Siwik and Mozgowoj*** 2015 [19] have presented a framework that relies on biometric authentication to produce digital signature. However, in this method the private keys of the user are kept in the same server and never forwarded even to the owner of the key. Alternatively, to sign a document, the user sends the document to the server, which signs the document using the keys stored for the user after authenticating the user using biometric authentication. In addition to the risk imposed by storing the private keys, this method requires intensive bandwidth, compared the standard digital signing method that communicates only the private keys of the user.

4- ***S . Haji and A. Varol*** 2016 [22] propose a real-time application, based on Windows operating system, for face recognition. The system is based on Eigen and Local Binary Patterns to measure the similarity between face images, to authenticate users, to reduce the effect of different illumination conditions. Despite the high recognition accuracy of 90%, the study shows that the accuracy is dramatically affected by any changes to the conditions of the environment, such as distance between the individual and the camera or ambient lighting, or when the images are collected using different cameras.

5- ***Rahmawati et al.*** 2017 [20] have used the smartphone's fingerprint authentication sensor to restrict access to the private keys of the

users, so that, each user can only access his/her private key. Despite the high accuracy of fingerprint-based physiological biometric authentication, the storage of the private keys in a server that is connected to a network presents a threat toward the security of these keys. An intruder that gains access to the database, without using the designed authentication method, can produce false signatures using any user's private key.

6- ***Lozhnikov and Sulavko*** 2018 [21] have proposed a method that allows users to digitally sign documents by using features that are extracted from their face and keystrokes of the keyboard. Despite the low error rate in this method, the study does not evaluate the security of the system, in terms of producing false signatures by intruders. Unlike physiological biometric features, the behavioral features extracted from the keystrokes are easier to be replicated by intruders.

1.3 Problem Statement

The number of digital documents being communicated over networks is rapidly increasing in recent years, according to the ease of access to computers and the internet. This increment imposes the need to validate the authenticity of received documents, especially with the increasing number of attacks being executed against these documents to manipulate their contents. Hence, digital signatures are being widely used as a proof of authenticity and sender's approval to the contents of the documents. However, the existing techniques rely on storing the private keys of the users, which are used to produce the digital signature in different locations, such as databases. Despite the use of secure schemes, such as biometric authentication, to restrict access to these keys, the security of the storage technique can still impose a thread toward the

system as gaining access to the stored keys allows attackers to produce any signature for any document. These authentication methods do not require or check the existence of the user in real-time during the signature production. Such a limitation reduces the security of these systems by allowing other users to attempt authenticating as legitimate users who are not present in real-time. Additionally, face biometrics are widely used to authenticate users into systems according to the high availability and robustness of facial features. Hence, the proposed method uses deep neural networks to extract the private keys of users based on facial features collected in real-time

1.4 Aim of the Thesis

The aim of this thesis is summarized as follows:

- Design a novel technique to produce the private keys of the users from real-time biometric information collected from them.
- The proposed method uses artificial neural networks that use 3-dimensional representation of the face images to produce robust and unique keys for each user.
- The extraction of the private keys directly from the biometric features significantly increases the security of the system, as these keys are never stored anywhere in the system.

1.5 Outline of Thesis

The rest of this thesis is:

Chapter Two: Theoretical Background

This chapter gives the background and review of the basis for algorithms and techniques that are used in this thesis.

Chapter Three: The Proposed System

This chapter describes the proposed system signing digital documents using facial features with their design and implementation.

Chapter Four: Experimental Results and Performance Evaluation

This chapter explains the results that have been got from the proposed system with discussion.

Chapter Five: Conclusion and Future Work

This chapter presents the conclusions about this work. Also, the suggestions for future work are given in this chapter.

Chapter Two

Chapter Two

Theoretical Background

2.1 Introduction

With the various range of applications that use digital information, the security of this information has emerged, such as authentication, integrity, and non-repudiation. Hence, digital signatures have been presented to guarantee these characteristics of the received information. Thus, the security of digital signatures has attracted significant attention, as they prove the identity of the data owner and their approval to the contents of the received document[25]. Additionally, biometric features have also been widely used to identify individuals when accessing digital services. Several applications, such as financial services and information security, have employed biometric features in one-to-many matching methods. These methods compare the features collected from the user to many features stored in the database in order to recognize and authenticate that user. Depending on the type of features being collected from the individual, biometric methods can be categorized into physiological and behavioral. This chapter presents the existing techniques being used in generating digital signatures and the use of biometric features to recognize system users[26].

2.2 Digital Signature

As digital signatures are generated to prove the identity of the user who is the source of the digital document and his approval to the contents of the document, each signature must contain unique information about the user and the document. Hence, the generated signature can be used to validate the received digital document, so that any change in the contents of the document produces a different signature than that attached with the document. Thus, digital signatures require two main techniques a hashing function and a public key encryption

method. The hashing function generates a unique value for the digital document which changes even if a single value changes in that document, and the public-key encryption method is used to prove the identity of the sender[27].

The fundamentals associated with digital signature are explained below :

2.2.1 Cryptography

Converting the message representation from plaintext to cipher formation so that, only the designated received can retrieve the original contents of the message is known cryptography. Several techniques have been proposed in this field of study which are characterized by using three dimensions the operations used to change the plaintext into the cipher message, number of keys required by the technique, and the approach used to retrieve the plaintext from the ciphered message. However, there are two main categories for these techniques, which are the symmetric, i.e. private-key, and asymmetric, i.e. public-key, encryptions [28].

2.2.2 Types of Cryptography

A. Symmetric Key Cryptography

Encryption and decryption methods in this type of cryptography require the same key, so that both the sender and the receiver must share the key prior to communicating the information. This type of cryptography can further be classified into block and stream cipher. Block cipher methods process the plaintext input as blocks, unlike stream cipher methods, which process characters in the input individually [28] .

B. Asymmetric Key Cryptography

The keys used by the encryption and decryption techniques in these cryptography methods are different, so that the key used for one task has different values than the key used for the other. Hence, these techniques provide more stability, as the private key is not revealed to any other users of the cryptography system. [28].

2.2.3 Factors of Digital Signature

There are certain factors that can affect, or describe, the performance of a digital signature method. These factors are the privacy, authenticity, integrity, and non-repudiation[29].

A. Privacy

This factor represents securing the information against any manipulation or unauthorized access by a third party . Hence, the transactions between businesses are not interfered by those parties [29].

B. Authentication

The authenticity of the information is proved by proving its source, so that the received data is guaranteed to be from the real sender, other than third-party intruders. This factor is the key toward building trust between the communicating parties, so that any alternation with the data can be detected by those parties when communicating over a network[29].

C. Integrity

Protecting the data from any accidental, uncontrolled, or unauthorized alternation during transmission is also an important factor when establishing communications over a network. The integrity of the data ensures that the received information is identical to the transmitted, otherwise, any alternation

can be detected. This allows the receiver to neglect any information if its integrity is found questionable[29].

D. Non-Repudiation

This factor represents the ability of the digital signature to prove the identity of the information origin. When this information is delivered, the receiver can identify whether it is coming from the sender it pretends to be originated from or not. Thus, the receiver can prove that the information they received is guaranteed to be from that sender, so that, the sender cannot repudiate sending such a message [29].

2.3 Digital Signature Algorithm (DSA)

The DSA is developed in 1991 by the National Institute of Standards and Technology (NIST) to be used in Digital Signature Standard (DSS). This algorithm is based on the problem of difficult computing discrete algorithm. DSA is mainly used to authenticate digital signatures in addition to their integrity verification. Using the Secure Hash Function, DSA generates and verifies the digital signatures, where the sender's private key is used to encrypt the hash generated for the document and generate the signature. Upon the arrival of the message, the receiver uses the senders, public key to decrypt the signature and retrieve the hash of the document sent by the sender. DSA also maintains compatibility with signing and verification functions [30]. There are three important processes in the DSA, which are the key generation, signing, and verification [31] .

Algorithm (2.1): Digital Signature Algorithm [31]

Input: p,q,g

Output : x,y, digital signature

Begin

Step 1: Choose prime number p and q, which in this case (p-1) mod q = 0.

Step 2: Count $g = h^{(p-1)/q} \bmod p$, which in this case $1 < h < p - 1$ and $h^{(p-1)/q} \bmod p > 1$.

Step 3: Specify the private key x, which in this case $x < q$.

Step 4: Count public key, $y = g^x \bmod p$.

Step 5: choose message to sign

Step 6 :Generate number k randomly for each message, where $0 < k < q$.

Step 7: Count $r = (g^k \bmod p) \bmod q$

Step 8: Count $s = (k^{-1} (\text{SHA-2}(m) + x*r)) \bmod q$, where SHA-2 (m) is SHA hash function to m message.

Step 9 : The digital signature is (r, s)

Step 10 :Count $w = (s)^{-1} \bmod q$

Step 11: Count $u1 = (\text{SHA-2}(m)*w) \bmod q$

Step12: Count $u2 = (r*w) \bmod q$

Step13: Count $v = ((g^{u1}*y^{u2}) \bmod p) \bmod q$

Step14: The digital signature is valid if v = r

END .

2.4 Cryptographic Hash Function

Cryptographic hash functions are used to produce fixed-length string referred to as the hash code for each variable-length input message. This code is appeded to the message upon transmission, where the message is assumed to be valid at that point. Hence, by calculating the hash of the received message and comparing it to the hash attached to the transmitted message, the reciever can validate the received message by simply comparing these hashes. To produce strong hash codes, a cryptographic hash function H is required to be collesion, preimage, and second preimage resistant [32].

2.4.1 SHA-2 Hash Function

The National Institute of Standard and Technology (NIST) defines a set of cryptographic hash algorithm in the Secure Hash Standard (SHS), which contains several Secure Hash Algorithms (SHA). Excluding the SHA-1, SHA-2 is introduced in 2001 and contains three hash functions, which are then updated to six functions in a subsequent update from the same institute. The resulting set of secure hash algorithms is widely used in different applications, according to their good performance compared to other hash functions. The different functions in the SHA-2 are distinguished based on the length of the hash code they produce, where the SHA-256 and SHA-512 are two of the popular variations among the others, such as the SHA-224, SHA384, SHA-512/224 and SHA-512/256. The basic two hash functions, i.e. SHA-256 and SHA-512, use similar mathematical operations to produce the hash code. However, the SHA-512 processes 64-bit words as its input, unlike the SHA-256, which uses 32-bit words as its input. Additionally, these functions have different variables initialization procedures, as well as some different values and parameters. The other variations rely on these two basic algorithms and truncate the outputted hash code into the required length. Hence, these two basic variants can be used to describe all the functions in the standard [33]. We use secure hash function, such as, SHA256 as a strong one-way function

.It is easy to compute its output value (hash), but is not possible (impractically difficult) to reverse the operation. This function are secure till date against different attacks, including collision attacks. Therefore, our selection of SHA256 as strong one-way function is completely safe [34].

2.4.1.1 SHA-256 Algorithm

To calculate the hash code of a message, M, the SHA-256 requires an input of length congruent to 448mod512. Thus, when an input with a length that does not satisfy this requirement is inputted to the algorithm, the message is padded until the required length is satisfied. The padding starts by setting the first bit in the pad, i.e. a value of one, then append a series of zeros until the required length is met. Then, a 64-bit block is appended to the message to produce N blocks, each of 512-bit length, i.e. $M^{(1)}$,$M^{(2)}$,.., $M^{(n)}$. An initial hexadecimal hash code is produced by the algorithm, $H^{(0)} = IV$, which consists of eight 32-bit words. The value of this initial hash is updated by feeding the input message in a schedule of sixty-four 32-bit words, which are denoted as W_0,W_1,…,W_{63}. The main steps executed by the SHA-256 algorithm to produce the hash code are shown in Algorithm (2.2) [32].

Algorithm (2.2): SHA-256 Algorithm [32]

Input : message

Output: hash of message

Step 1 :Prepare the message W_t

$$W_t = \begin{cases} M_t^{(t)} & 0 \le t \le 15 \\ \sigma_1^{\{256\}}(W_{t-2}) + W_{t-7+}\,\sigma_0^{\{256\}}(W_{t-15}) + W_{t-16} & 16 \le t \le 63 \end{cases}$$

Where

$$\sigma_0^{\{256\}} = \mathrm{ROTR}^7(\mathrm{X}) \oplus \mathrm{ROTR}^{81}(\mathrm{X}) \oplus \mathrm{SHR}^3(\mathrm{X})$$

$$\sigma_1^{\{256\}} = \mathrm{ROTR}^{17}(\mathrm{X}) \oplus \mathrm{ROTR}^{19}(\mathrm{X}) \oplus \mathrm{SHR}^{10}(\mathrm{X})$$

Step 2: Initialize the eight working variables a, b, c, d, e, f, g, and h, with the (i-1)st hash value

Step 4 : For t= 0 to 63

{ $T_1 = h + \sum_1^{\{256\}}(e) + Ch(e,f,g) + K_t^{\{256\}} + W_t$

$T_2 = \sum_1^{\{256\}}(a) + Maj(a,b,c)$

H= g, g = f, e = d + T_1 , d = c, c = b, b = a, a = $T_1 + T_2$

}

Step 5: Where $Ch(x,y,z) = (x \wedge y) \oplus (x \wedge z) Maj(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$

$K_t^{\{256\}}$ is a sequence of sixty-four constant 32-bit words

$$\sum\nolimits_0^{\{256\}}(x) = \mathrm{ROTR}^2(x) \oplus \mathrm{ROTR}^{13}(\mathrm{x}) \oplus \mathrm{ROTR}^{22}(x)$$

$$\sum\nolimits_0^{\{256\}}(x) = \mathrm{ROTR}^6(x) \oplus \mathrm{ROTR}^{11}(\mathrm{x}) \oplus \mathrm{ROTR}^{25}(x)$$

Step 6: After repeating steps one through four a total of N times (i.e., after processing M(N)), the resulting hash functions is $H_0^{(N)} \parallel H_1^{(N)} \parallel H_2^{(N)} \parallel H_3^{(N)} \parallel H_4^{(N)}\, H_5^{(N)} \parallel H_6^{(N)} \parallel H_7^{(N)} \parallel$.

2.5 Biometric

The term "biometric" is a combination of the two Greek words, bio and metric, which mean life and measure sequentially. Verifying individuals automatically based on specific biometric features is known as biometric verification. These features are collected from physiological or behavioral characteristics of that individual. The use of these features has significantly better capabilities in distinguishing legitimate users from intruders, compared to the use of something you know, e.g. passwords, or something you have, e.g. ID cards. Accordingly, biometric verification has been widely used in different applications in recent years, such as ATMs, mobile phones, security installations, computers , etc [35]. Biometric verification systems rely on pattern recognition techniques, which extract features from the templates collected from the individual. These features are compared to other sets of features that are previously collected from legitimate users, in order to evaluate the authenticity of the user being verified [36].

2.5.1 Types of Biometrics

There are several types of biometric, but five of them are commonly used according to the better security they provide. Basically, recognizing an individual using unique features collected from the human body, such as palm prints, retina, face, voice, iris and fingerprint, is the aim of biometrics. This technology has been employed to secure the users information and devices by ensuring the denial of any unauthorized users' access. Hence, gaining physical access to such devices denies the intruders digital access to them, which protects the important and sensitive information being stored on such devices [37].

2.5.2 Biometric Recognition

Several characteristics that exist in the human body can be used to recognize and distinguish individuals. The use of each set of the features extracted from these biometrics has its own strong and weak points. Hence, the type of the biometric

features collected from the human are selected based on the requirement of the application the employs these features and how suitable are the properties of the selected biometrics. These characteristics can be categorized into two main classes [38, 39]:

A. **Physiological** are extracted directly from a certain part of the human body that is unique for each individual, such as iris, fingerprint, and face. The features extracted from such biometrics are more robust and faster to collect but may require more expensive equipment, such as fingerprint or iris scanner.

B. **Behavioral** are collected from the unique way by which an individual executes a certain action or task, such as gait, signature and keystroke patterns. Figure (2.1) presents different biometric types, which generally belong to one of two categories: (a) physiological which is associated with the body modality & (b) behavioral which is associated with the person's manner [36,40].

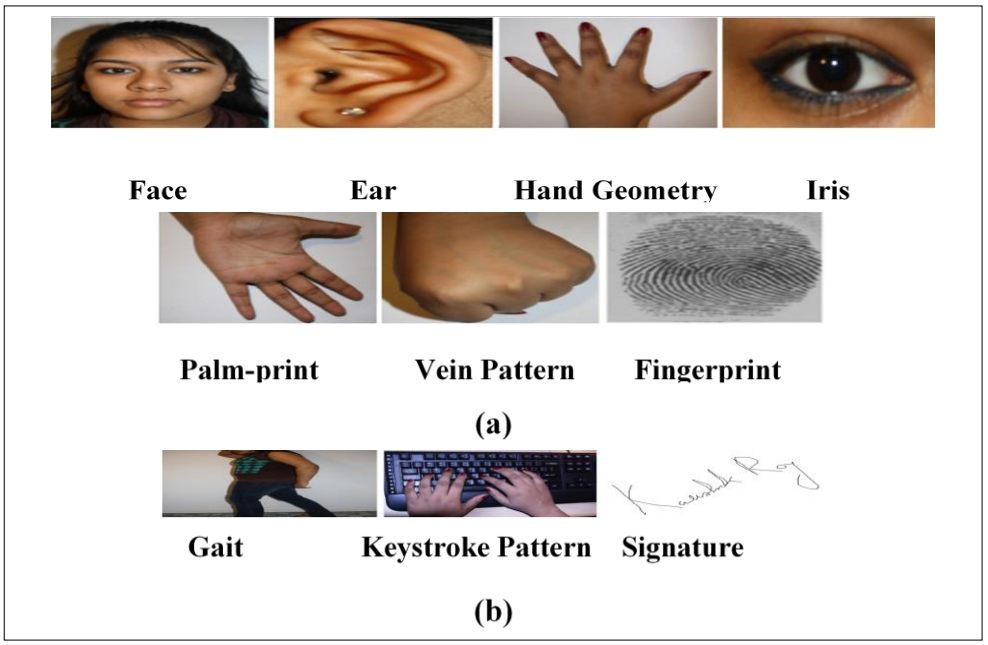


Figure (2.1): Examples of different biometric traits: (a) physiological traits, and (b) behavioral traits .

2.6 Face Recognition

The use of facial features collected from an image that contains a face template collected from a certain individual is known as face recognition. Face recognition techniques rely on the use of computer vision and pattern detection methods, so that the identities of the individuals that the face images are collected from are recognized. Several applications have been developed based on face recognition, such as medical science, robot intelligence, video surveillance, and criminal investigation. Compared to other biometric features, face recognition requires no expensive equipment, as images can be collected passively using a digital camera, and can provide distinctive information for the applications that employ these techniques. Depending on the employment of the facial features in the application, these features can be used in two approaches:

1) Face authentication
2) Face recognition

Face authentication is the process of matching the queried face with a given dataset of images to authenticate the claimed identity. Face recognition is the process where facial features are compared to those stored for the authentic individual that the collected features are supposed to be for [41].

2.7 Real-Time Face Recognition System

A face recognition system is implemented mainly by using several modules such as : face localization (Detection), normalization, feature extraction, and matching. The first two modules preprocess the input image that contains the face, so that the remaining two steps can recognize the individual in that image [21].

2.7.1 Face Detection and Cropping

Images collected from the users cannot be guaranteed to contain only the face image, according to the different possible poses and positioning of the individuals. Hence, an important step to improve the performance of the face recognition system is to detect the exact position of the face and crop out the remainder of the

image. Using such procedure, less unrelated features are forwarded to the feature extraction step, which can significantly improve the performance of the matching step. Such goal is achieved by cropping the image to the boundaries where the face is detected to be in [42]. Multi-Task Cascade Neural Networking (MTCNN) is one of the recent techniques that have shown outstanding performance in localizing the faces in an image .

2.7.1.1 MTCNN Based Face Detection

In order to detect face landmark for face detection, the Multi-task Cascaded Convolutional Neural Networks (MTCNN) algorithm uses three neural networks, namely, the p-net, r-net, and o-net. The first stage, i.e. the proposal network, predicts bounding boxes that are candidates to have face images in them based on detected face landmarks, similar to the Faster R-CNN's attention network. The face is defined using five landmarks, two that represent the eyes, one for the nose, and two define the mouth. Overlapping regions are summarized by selecting the larger region as the potential region to contain face image. Next, the neural network in the second stage, i.e. the refine network, refines the output of the p-net by eliminating the regions that have less face-like patterns. Then, the same processes of joining overlapping regions and eliminating less face-like regions are repeated for the output of the r-net using the output neural network. The MTCNN has shown better performance than other face detection methods, including those that use convolutional neural networks, while maintaining very low execution time, which makes suitable for real-time applications [43]. The popular CNN-based descriptor generation methods, which generate a fixed size vector for each input face, as the Facenet .

2.7.2 Image Resizing

According to the possibly different poses and positions of the users during the collection of the face image, the size of the region detected by the MTCNN to include a face image is unknown. Hence, the size of the cropped image cannot be fixed. To maintain a fixed image size in further processing, the images cropped based on the prediction of the MTCNN are resized using bilinear interpolation [42].

2.7.2.1 Bilinear Interpolation

When an image is resized, the pixels in the destination image are mapped into the original image, being resized. When a pixel is mapped on an exact location of a pixel in the source image, the value of that pixel is placed in the destination image. However, if the mapped pixel does not fall on an exact pixel, the value in the destination image is selected based on the most adjacent, to the mapped position in the source image. Linear interpolation uses the two most adjacent pixels, which can be either on the *x* or *y* axes. For more accurate values, bilinear interpolation uses the adjacent pixels in both axes to calculate the value of the pixel depending on its mapped position among the four pixels in the source image. Thus, the bilinear interpolation provides better resized images, compared to the linear interpolation, while maintaining low computational complexity [44] .

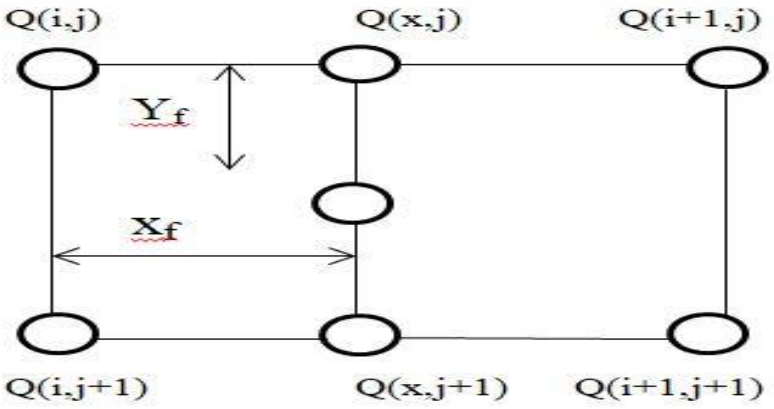


Figure (2.2): Bilinear Interpolation[39]

Considering four neighboring pixels, Q(i,j), Q(i+1,j), Q(i,j+1) and Q(i+1,j+1), of a position mapped in a source image with size of $M \times N$, i.e. =[0,1,2,…M] and j=[0,1,2…N], the temporary pixel created by the vertical and the horizontal direction is as shown in equation (2.1) and (2.2) [42].

$$P(x', j) = (1 - x_f) * P_{(i,j)} + x_f * P_{(i+1,j)} \quad (2.1) .$$

$$P(x', y') = [(1 - x_f) * P_{(i,j+1)} + x_f * P_{(i+1,j)}] * (1 - y_f) + [(1 - x_f) * P_{(i,j+1)} + x_f * P_{(i+1,j+1)} y_f \quad (2.2) .$$

Where x_f is the scale parameter in the horizontal direction and y_f is the scale parameter in the vertical direction

Algorithm (2.3): Image Resizing Algorithm

Input: Image and required dimensions
Output :resize image
Begin
Step 1: Read the original image and required dimensions.
Step 2: Create a blank image with the required dimensions.
Step 3: For each pixel in the blank image
Calculate a new value using equation (2.1)and (2.2).
Step 4: Return resized image .
End.

2.7.3 Standardization (Normalization)

According to the variation in the illumination of the environment that the face images are collected from and the different poses and positions of the user, the range of values in the image can be different from one shot to another. For instance, the intensity values of images collected for the same user can be of different ranges depending on the lighting at the instance for which the image is taken. However, the difference in contrast among the different facial features is relatively similar. Thus, the values are normalized to maintain a constant range of

intensity values in the image, regardless of the conditions that the face image is collected within. Such normalization can be achieved by using Equation (2.3) [45] .

$$Xstd = \frac{x - x\,\text{mean}}{\sigma x} \quad (2.3)$$

where, x_{mean} represents the mean of the corresponding feature data column x, and σ_x denotes the standard deviation [46] .

2.7.4 Face Feature Extraction

In order to match face images, it is important to produce a feature vector that has the ability to represent the distinguishable features in the face. This vector can then be used to measure the similarity between two face images. The aim of producing such vector is to emphasize intra-class features, i.e. features that can distinguish one user from another, and neglect inter-class features, i.e. features that can be variable for the same individual [21].

2.7.4.1 CNN Based Face feature extraction

Convolutional neural networks have shown the best performance among other types of deep learning methods when interacting with images as input to the neural network. Moreover, unlike traditional machine learning techniques, the characteristics of the features extracted by neural networks are recognized during training, i.e. not predefined. Convolutional layers contain multi-dimensional filters that are trained to detect local features, according to the number of dimensions that these filters have. Thus, using such layers, the neural network can detect multi-dimensional features regardless of their position in the input.

Several type of layers can be used in a neural network, depending on the output required from that layer. These types of layers are the convolutional layers, pooling layers, and fully connected layers. The convolutional layers detect the local features, whereas the pooling layers summarize the input and reduce its size to reduce complexity. The fully connected layers, also known as dense layers, are used to combine the detected features and produce the output required from the

neural network. In a classification application, for example, the output layer contains a number of neurons equal to the number of classes in the domain, so that each neuron represents a class. Moreover, the dimensions of the input to the convolutional layer are defined by the type of the input provided to it. For instance, colored images are represented mathematically using three-dimensional array. The length and width of the layer represent the size of the image, while the third dimension is used to hold the pixel values of the Red, Green, and Blue components [47] , see figure (2.3).

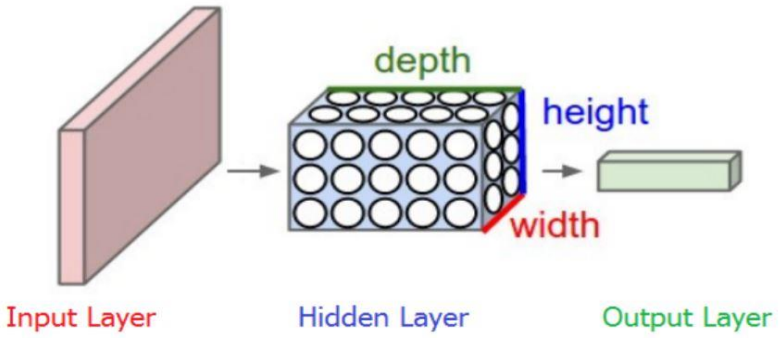


Figure (2. 3) : A convolutional neural network with three-dimensional input [47].

To provide the neural network with nonlinearity, activation functions are used to process the output of neurons before being forwarded to the next stage, which can be another layer of the output. A CNN consists mainly of several types of layers, as shown in the example in figure (2.4). These layers are:

1. Convolutional Layer (CONV): It contains the filters that represent the weights of a certain neuron in the convolutional layer. The output of the neuron is also a multi-dimensional array, depending on the number of dimensions in its input.
2. Rectified Linear Unit Layer (ReLU): This type of activating functions only allows positive values to be forwarded to the next stage, which has shown significantly faster training and better performance.
3. Pooling Layer (POOL): It down-samples its input in a nonlinear approach, so that only important features are maintained, depending on the type of features required in the application.

4. Fully Connected Layer (FC): It produces one-dimensional vector that can be delivered to another layer to detect more complex features, or as the output of the neural network [46].

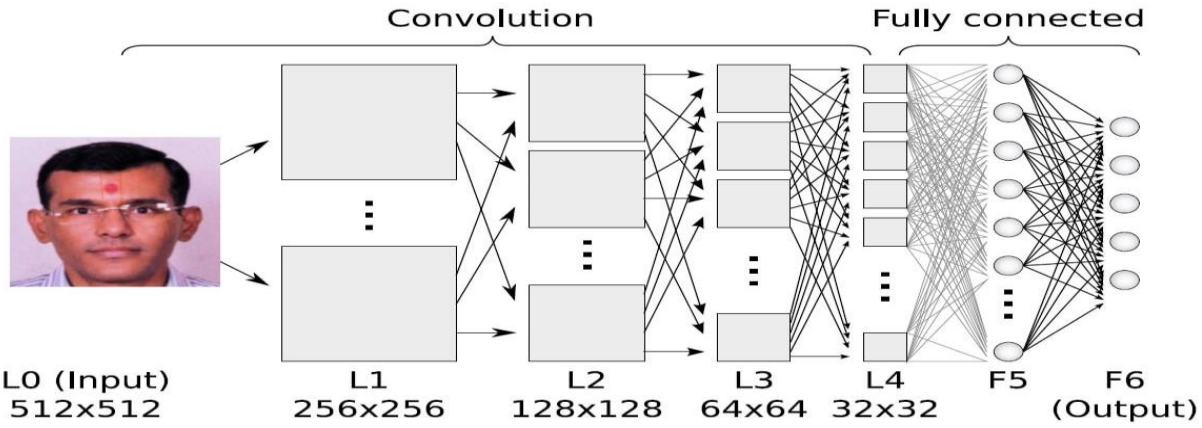


Figure (2.4) : A sample convolutional neural network[47]

2.7.5 Face Matching

The features extracted from the face image are used to match the identity of the individual in the face image. This matching can be executed against a single descriptor collected from the authentic user. In such a case the output of the face matching technique is a binary decision,either to match or not. Another approach matches the features to set of descriptors of multiple individuals, in which the output represents the similarity measure between the input feature vector and each template in the database. Finding a suitable similarity measure is the main challenge in face recognition, according to the variations that can exist in the collected face images [21].

2.7.5.1 FaceNet

Proposed by Google researchers in 2015, FaceNet is a deep convolutional neural network that has shown effective solution to the face verification hurdles. Similar to word embedding, this neural network outputs a fixed-size, i.e. 128-feature, descriptor that represents the input face image in the Euclidean domain. To train

the neural network to output such values, the developers have used triplet loss, which uses a positive and negative sample images for each anchor image. The embedding layer is placed prior to the output layer, which is the layer that uses the tripled loss. However, as the values outputted by the neurons in the output layer are calculated based on the outputs collected from the embedding layer, the values outputted from the neurons in the embedded layer must be similar for positive samples and different for negative ones. As the values required from the FaceNet are satisfied by these characteristics, the output layer is then eliminated from the model and the values outputted from the embedding layer are collected as the output of the FaceNet . Figure (2.5) shows the block diagram of the FaceNet architecture[43].

Figure(2.5): Facenet Block Diagram [43]

Regardless of the visual similarity between the images, the triples loss relies on the actual person in the face images to calculate the loss during the training of the neural network. An anchor image represents the image being paired with other face images in the training dataset, which can be of any individual in that dataset. The positive sample is another face image collected from the training dataset for the same person in the anchor image. Additionally, the negative sample is a face image collected from the dataset for a different individual. Using such approach, the neural network can then recognize the inter- and intra-class variations, so the similar values are produced for the same individual by neglecting the intra-class variations. Moreover, by emphasizing the inter-class variations, the neural network can produce different vectors for different individuals according to the use of the triplet loss, as shown in Figure(2.6) [43]

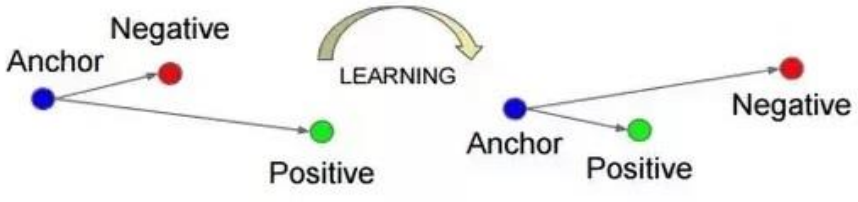


Figure (2.6): Triplet Loss Training [38]

2.7.5.2 One Shot Learning

Normally, training a machine learning technique requires providing samples of all the possible outputs to be expected from that neural network. Such training allows the neural network to learn the features that characterize each output and use it to predict it. However, in some applications, it is impossible to collect training data for all the output required from the neural network. For instance, in face recognition, it is impossible to collect sample images for each individual in the world. Hence, one-shot learning is used to allow the neural network to recognize the features that define the characteristics of the inputs in a certain domain. Then, this knowledge is used to process future inputs that have not been included in the training.

Siamese networks are neural networks that share the same weights, i.e. produced using a single training procedure. The aim of these networks is to maximize the distance between their outputs when the inputs are from different classes in the same domain. However, these outputs are required to be very similar when the inputs are of the same class, regardless of the intra-class variations that can be present. Hence, the similarity between the inputs, or the probability that those inputs are form the same class, can be predicted by calculating the distance between the vectors created by the Siamese networks. As shown in Equation (2.4), the embedding of inputs x_i and x_j and the convolutional function with the max-pooling fw(·) produce a single-dimensional vector. By processing this vector by using the neurons in the output layer, i.e. g(·), the distance between these inputs can be calculated. Hence, inputs with less distance between them are expected to be more similar and vice versa [43].

$$dist(x_i, x_j) = | g(f_w(x_i)) - g(f_w(x_j))| \quad (2.4)$$

2.8 semi-supervised learning (SSL)

Semi-supervised learning describes a class of algorithms that seek to learn from both unlabeled and labeled samples, typically assumed to be sampled from the same or similar distributions. Approaches differ on what information to gain from the structure of the unlabeled data [48] .

2.9 Authentication Performance Measures

An authentication technique is required to provide a prediction for each user trying to login to the device, whether to be a legitimate user or an intruder so that access to the device and the information stored on it is granted to legitimate users, while an intruder's access is denied[49]. Thus, there are two types of users from the authentication technique's point of view, which are the legitimate users (L) and intruders (I). In order to evaluate the performance of an authentication technique, the predictions provided by the authentication technique are distributed against the actual state of the users in a confusion matrix similar to table (2.1), where *TL* represents the number of legitimate users that are predicted to be legitimate by the authentication technique, *FL* is the number of intruders predicted to be legitimate users, *TI* is the number of intruders that are correctly recognized by the authentication technique and *FI* is the number of legitimate users predicted to be intruders.

Table (2.1): The confusion Matrix used to Calculate Performance Measures of an Authentication Technique.

		Predicted	
		Legitimate	**Intruder**
Actual	**Legitimate**	TL	FI
	Intruder	FL	TI

The first performance measure of an authentication technique is: -

- False Acceptance Rate (FAR): It is the ratio of intruders predicted to be legitimate users, i.e. granted access to the device and to the total number of actual intruders in the evaluation. This measure represents how secure is the authentication technique, where lower FAR indicates higher security as the ratio of intruders that are allowed into the device is less[50]. The equation to calculate the FAR, using the confusion matrix shown in table (2.5), is shown in equation (2.5).

$$FAR = \frac{FL}{FL + TI} \tag{2.5}$$

- False Rejection Rate (FRR): It represents the ratio of legitimate users that are denied access to the device. This measure describes the usability of the authentication method, where lower FRR represent better usability as the higher rates indicate that more legitimate users fail to authenticate into their devices. Equation (2.6) shows the calculation required to compute the FRR using the confusion matrix shown in table (2.1).

$$FRR = \frac{FI}{TL + FI} \tag{2.6}$$

- Equal Error Rate (EER): By adjusting the value of the threshold used to make a decision, whether to authenticate the user or to deny access, depending on the calculated probability of being legitimate user, it is possible to adjust the FAR and FRR. Increasing the threshold requires higher matches between the current user and the template stored for the legitimate user, which reduces the FAR. However, such increment also increases the FRR, as the input of the legitimate user is very restricted to match the stored template. At a certain threshold, the FAR equals the FRR, where that FAR or FRR value is selected as the EER [51], as shown in Figure (2.7).

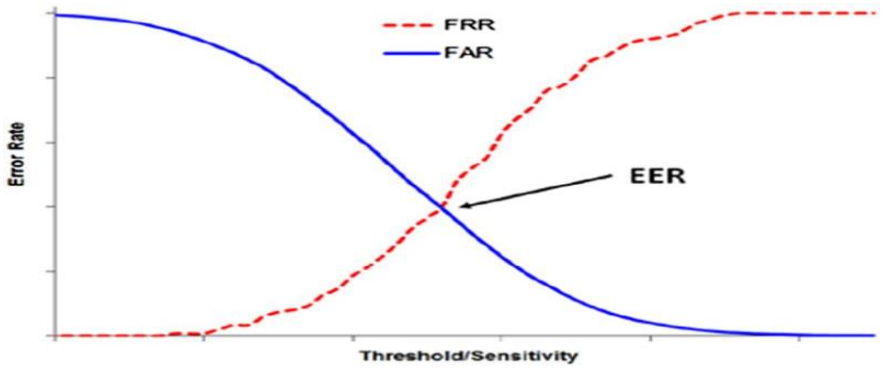


Figure (2.7) : The relationship between the FRR, FAR, and EER[1].

- Accuracy: The accuracy of the predictions provided by the authentication method is measured as the ratio of the correct predictions, regardless of the value of the predictions to the total number of predictions. The accuracy is calculated as shown in equation (2.7) [51].

$$Accuracy = \frac{Tl + TI}{TL + FL + TI + FI} \tag{2.7}$$

- Precision: Per each class, the precision measure describes the ratio between the number of data instances that are correctly predicted to be in that class to the total number of instances predicted to be in it. Hence, the precisions of legitimate and intruder classes are calculated as shown in Equations (2.8) and (2.9), respectively[52].

$$Preicion_{legitimate} = \frac{TL}{TL + FL} \tag{2.8}$$

$$Preicion_{intruder} = \frac{TI}{TI + FI} \tag{2.9}$$

- Recall: Represents the ratio of the correctly predicted instances at a certain class to the total number of instances that are actually in that class. Accordingly, the recall values of the legitimate and intruder classes are calculated as shown in Equation (2.10) and (2.11), respectively [52].

$$Recall_{legitimate} = \frac{TL}{TL + FI} \tag{2.10}$$

$$Recall_{intruder} = \frac{TI}{TI + FL} \quad (2.11)$$

- F1-Score: Summarizes the precision and recall measured into a single value that can be used to describe the overall performance of the classifier[52]. Thus, this measure is calculated using the precision and recall values as shown in Equation (2.12).

$$F1\text{-}Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (2.12)$$

2.10 Popular Authentication-Related Threats

There are two types of security attacks: human-based and technology based. With the human-based attacks, an adversary will interact with the target (victim), who has valuable information, such as through social engineering attacks. On the other hand, with technology–based attacks, an adversary can gain access to secret information by using non-interactive means, for instance, a phishing email . Password attacking involves different character combinations being tried until a match with the correct password is found. There are many types of password attacks, some of the most important ones are described next . In this section, the popular threats are discussed, and then they will be performed on the new authentications techniques that have been proposed in this thesis to explain how they can hold out against these attacks [54].

2.10.1. Brute-Force Attack

In brute-force attack, the attackers attempt to input all possible passwords to mimic the user[54].

2.10.2. Dictionary Attack

A dictionary attack is carried out on verification data by trying out every word in the dictionary. This type of attack is targeted at sites with a high probability of

success, such as those with weak passwords or with only a few key combination numbers. This attack is quicker than one of brute force and is more successful when a weak commonly used or short password is used. However, when the password contains special characters, it becomes more complex to hack into web sites. Moreover, a dictionary attack is the commonest method for hacking password hashes. There is a wide use of dictionary words by attackers aimed at analysing passwords, which can quickly crack hashes. That is, this involves using either very big dictionary files that contain potential passwords in their millions or a combination of words in the dictionary. It works by calculating the hash value of every dictionary file password and using it in comparison with the hash value input of any unknown one. When a match is discovered between the dictionary text and the hash value,

the input password will be the same. Whilst this is a faster than other methods of attack, it is not as successful. It has a generally good success rate when used for common passwords, and that is why numerous passwords are continually cracked using this method by attackers[54].

2.10.3. Phishing Attacks

This is where an attacker attempts to retrieve legitimate users' confidential and sensitive credentials fraudulently by mimicking electronic communications from a trustworthy or public organization in an automated fashion. The aim of phishing is to steal sensitive information, such as online banking passwords and credit card information from Internet users. These attacks use a combination of social engineering and technical spoofing techniques that persuade users into giving away sensitive information that the attacker then uses to make a financial profit [54].

2.10.4. Shoulder-Surfing Attack

Shoulder-surfing attack occure when someone sees the inputted password over the shoulder of a person. There have been many attempts to resist this type of attack, including eye-gaze entry, tactile/haptic (vibration) patterns, and digitally signing in with pressure on a touch screen[54] .

2.10.5. Guessing attack

In guessing attack, the attackers guess some passwords using specific information of the user and utilize the passwords to get some critical information of the user [54].

2.11 3D Shapes

Recently, with the rapid development of 3D imaging and visualization techniques, 3D shape analysis and processing (e.g. surface registration, 3D shape retrieval) have been widely studied. As stated in, capturing 3D shape makes an extremely wide array of new kinds of application possible. 123 J Math Imaging Vis For example, virtual reality (VR), augmented reality (AR), 3D biometrics, 3D human–machine interaction, 3D medical imaging, 3D remote sensing, to name just a few. Behind these applications, a fundamental and challenging problem is how to represent and characterize the geometric structures of these discrete 3D shapes (surfaces or volumes). Up to now, many approaches have been proposed to solve this key issue. As a special application of 3D shape analysis and processing, the key issue of 3D face recognition has also been widely addressed. Facial surface measurements, i.e. points ,curves ,stripes , regions , normals , curvatures , geodesic distance , have been popularly used to represent and characterize 3D face shapes. Among them, surface curvatures, including Gaussian curvature, mean curvature, principal curvatures, as well as shape index values, are the most widely used ones . Thus, the representation and characterization of 3D face surface are the most important techniques for an efficient 3D face recognition system, where curvatures play fundamental effect [53].

Chapter Three

Chapter Three

The Proposed System

Signing Digital Documents Using Facial Features

3.1 Introduction

According to the security threats imposed by storing the secret key, the main aim of the method proposed in this thesis is to generate these keys using users' facial features. Hence, the physical presence of the user is required to generate the private key and sign the digital document, which increases the responsibility of the produced signature. Moreover, gaining access to the database of the system does not allow producing any false signatures when the proposed method is used, as the secret keys are never stored in it. To achieve such a task, the proposed system employs an artificial neural network that extracts the secret component of the DSA from the face image. However, as the face images may vary for the same individual, it is important to validate the keys produced based on the values extracted from the face image. This validation is required in two stages, user registration and digitally signing documents.

3.2 The Proposed System

The proposed system can be categorized into three main tasks. The first task is signing up users to the digital signature system, which requires a system administrator to register and manage the users. The second task is signing the digital documents, and the third task is validating the signature of the received signed document, which can be conducted by any of the users of the system. Figure (3.1) provides an overview of the three main tasks of the proposed system. Sections (3.2.1), (3.2.2), and (3.2.3) describe each of these tasks respectively in depth.

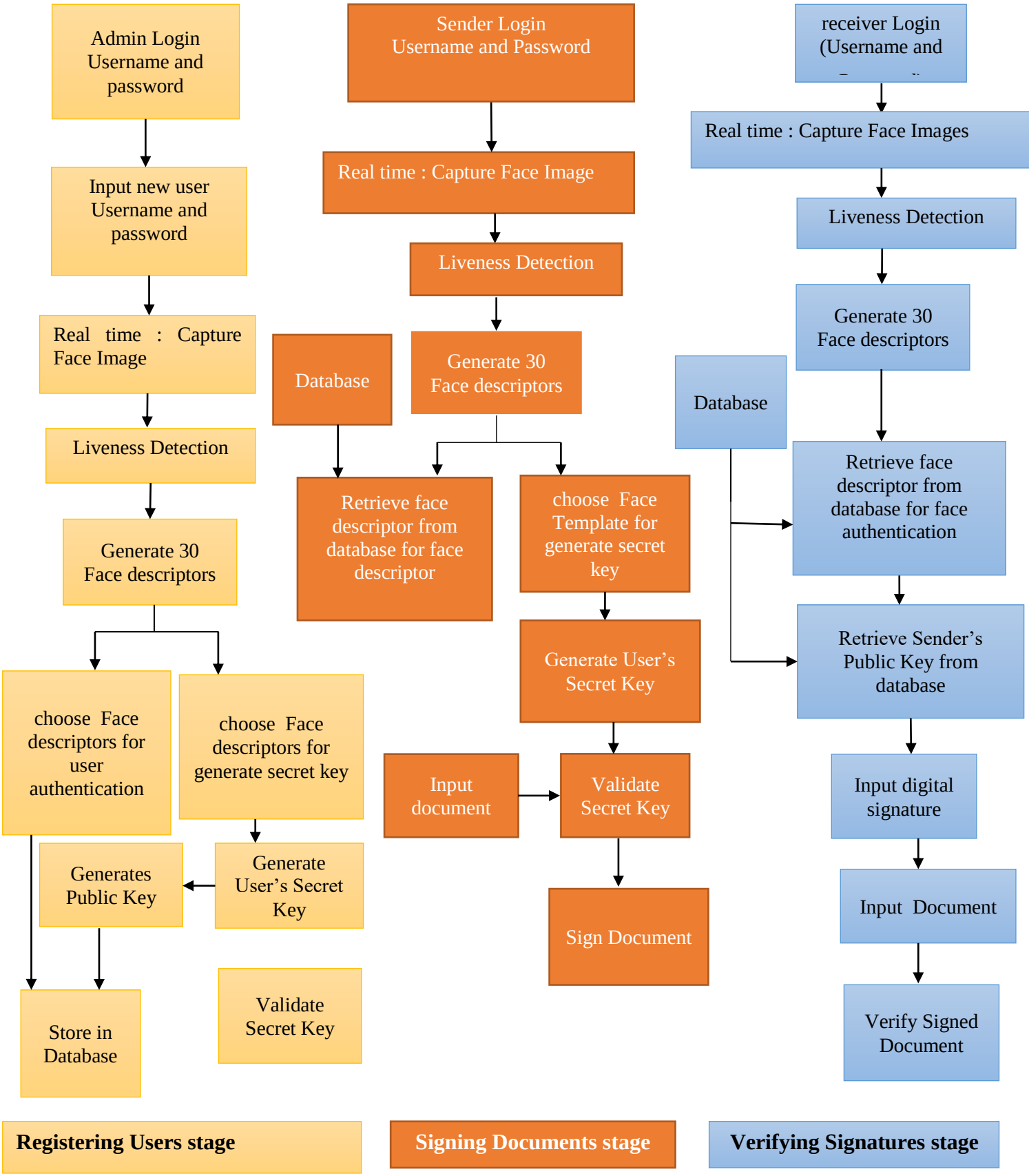


Figure (3.1): general block diagram of the proposed digital signature system.

3.2.1.Registering Users

Although registering users to a system may seem a straightforward task, the security measures considered in the proposed system require a more complex scheme, as shown in Algorithm (3.1).

Algorithm (3.1) : User registration procedure.

Input: **Output:**	Administrator's login credentials, Live video frames. New user's login credentials in the database.
Step1:	AU, AP ← Input administrator's username and password. If administrator is NOT authenticated as shown in algorithm (3.2.1.A): End //Quit the program
Step2:	UU, UP ← Input new user's username and password. HUP ← Calculate the hash of the user's password.

Step3:	C ← 0
	VF ← [(30×160×160)] //Empty array with (30×160×160) dimensions.
	While C<30:
	F ← Capture video frame.
	x, w, y, h ← Detect face image in F. // using MTCNN
	If x, w, y, h are not *null*:
	F ← F[x:x+w, y:y+h] //Extract the region that has the face image.
	F ← Resize F to 160×160. // using Bilinear interpolation
	VF[C] ← F
	C ← C+1 //Increase the counter by one.
	End If
	End While

Step4:	If liveness is detected in VF: //Section(3.2.1.B) FD ← Generate faces descriptors for VF. //using Facenet sk ← Generate DSA's secret key using FD. // algorithm (3.2.1.E) pk ← Generate user's public key. // using DSA FT ← Generate user's face template from FD.

	//Section(3.2.1.F) End If
Step5:	Store UU, HUP, pk, FT in the database. END

A. Administrator Login

The administrator of the system is responsible for managing the users of the digital signature system. Hence, adding users to the system is restricted to the administrator. The administrator is identified and authenticated using the username and password, where the hash of the password, calculated using SHA-256 hashing function, is stored in the database to maintain its security as illustrated in Section (2.4.1). The hash of the inputted password is calculated using the same hash function and compared to the hash of the administrator's password stored in the database. If the hashes are identical , the administrator is authenticated, otherwise, the password is considered invalid and the login process is terminated. This approach ensures authenticating the administrator without storing the password as plain text in the database, so that compromising the database does not reveal the administrator's password to the attacker. Algorithm (3.2) summarizes the administrator's login procedure.

Algorithm (3.2): Administrator's authentication procedure.

Input: **Output:**	Administrator's username and password. Administrator's login status.
Step1:	C ← 0 //Start a counter for the number of attempts
Step2:	AU ← Input administrator's username. AP ← Input administrator's password.
Step3:	PH ← Calculate AP's hash. // using SHA-256
Step4:	DU, DH ← Retrieve administrator username and password hash from the database.
Step5:	If AU==DU and PH==DH: Return True Else If C<3:

	C+=1 //Increment the attempts counter. Goto Step1 Else: Goto Step6 End If End If End

B. Liveness Detection

In order to avoid any spoofing attacks, using still images or videos recorded for the signing user, a neural network is trained to distinguish live videos from those produced in a replay attack. For this purpose, the neural network, shown in Figure (3.2), is implemented and trained using data collected from 33 volunteers.

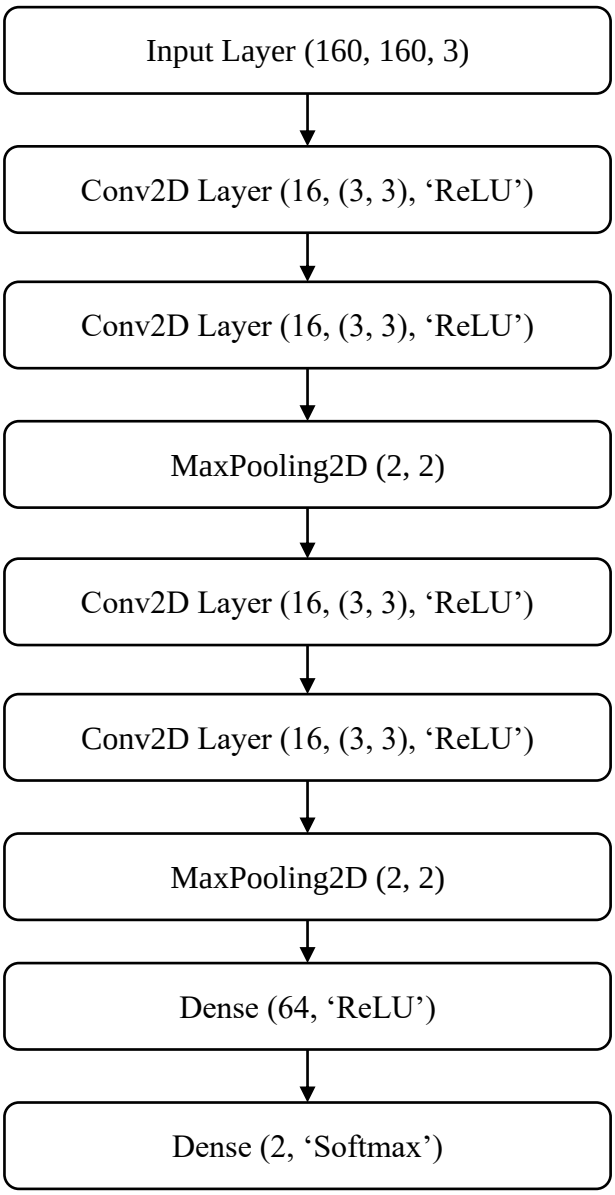


Figure (3.2): Liveness detection neural network.

As Algorithm (3.1) shows, the input of the liveness detection procedure is a set of 30 face images. The proposed liveness detection neural network provides the probability of each face image to be captured from a live face, instead of a replication of the actual user's face image. Depending on the performance evaluation, a threshold value is selected to provide a single prediction for the entire set based on the value collected for each face image. Moreover, to train this neural network, the videos collected from the volunteers are provided to the neural network in two modes. The first mode uses the frames captured directly

from the users using the camera, which represent the live examples to the neural network. To produce the spoofing attacks, i.e. replicating the users' face images, these videos are played back using a smartphone, instead of the actual user presence, and captured through the camera. Additionally, a single frame from the video is selected and displayed on the smartphone to simulate using a still image of the legitimate user. Face images collected from the frames of the latter videos or still images are used as examples of spoofing attacks, i.e. non-live face images.

C. Random Seeds Generation

According to the requirements of the DSA algorithm described in Section (2.3), the secret key requires one random number. To produce this number, a random generator is initiated using a seed value .The use of the same seed value produces the same private key. Thus, robust and unique, i.e. the same value is generated for the same user and different values are generated for different users, and seed is required for each user . The keys generated for each user are the same every time the user signs a document and different with each user. To extract such seed value from the face image of the user, a neural network is implemented to extract robust and unique values for each user, based on the descriptor generated by the FaceNet neural network. Therefore, a neural network is implemented for this purpose, as shown in Figure (3.3).

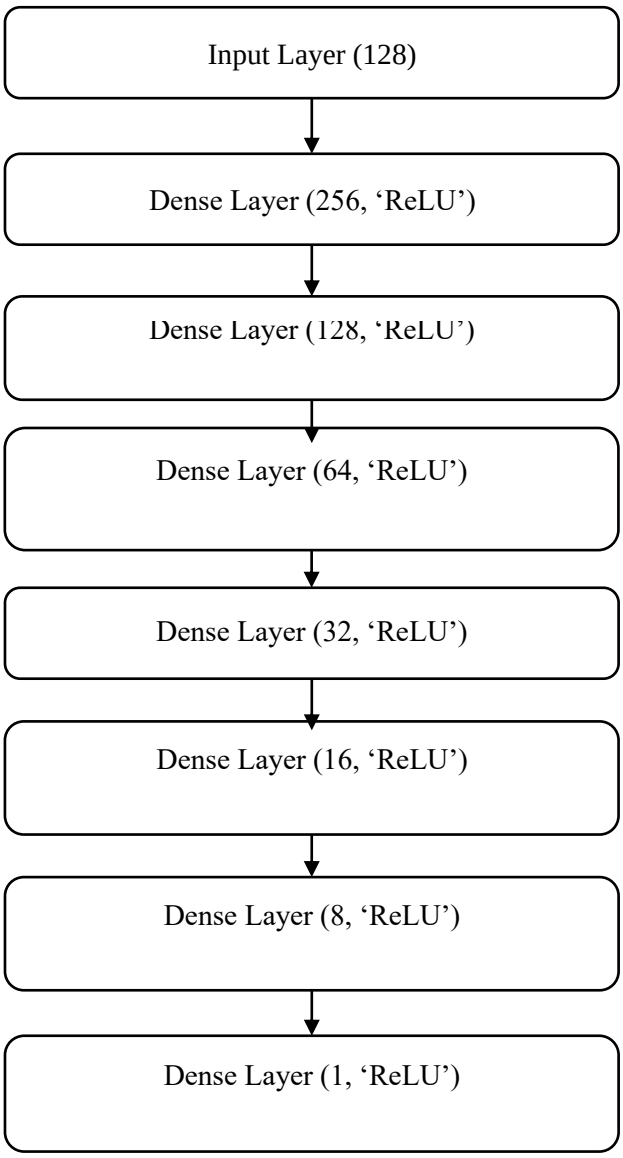


Figure (3.3): Random seed generation neural network.

As Figure (3.3) shows, the activation function used in the output layer is ReLU, which allows output values to be any value from zero to infinity. However, the values suitable for each face image are unknown and cannot be manually set because setting such values can be inappropriate for the computations in the neural network. Thus, a semi-supervised training approach is used to train this neural network, as shown in Algorithm (3.3).

Algorithm (3.3): Training random seed generation neural network.

Input: **Output:**	Face descriptors, Faces labels. A trained random seeds generation neural network.
Step1:	nn ← Create neural network shown in Figure (3.3).
Step2:	//Start assigning optimal values and training the neural network. T ← 0 //A counter for the training iterations.

Step3:	//Read input data
	F ← Read face descriptors.
	L ← Read faces labels. //The individual that each face image belongs to.
	G ← len(unique(L)). //The number of individuals.
	OL ← zeros(len(L)) //List of zeros identical to the labels.

Step4:	//Create the optimal distribution of the values.
	O ← [(G)] //Create a list with length equal to the length of the input data.
	S ← int(1000000000/G) //Calculate the step size between two consequent optimal values.
	D ← int((1000000000-S×G)/2)
	For i = 1 to G:
	O[i] ← i×S+D
	End For

Step5:	// Compute the centroid of the prediction per each user.
	preds = nn.predict(F) //Predict a value per each input image.
	C ← [(G)] //An empty array to store the centroid per each individual.
	x ← 0 //A counter
	For i in unique(L): //For each individual in the labels.
	C[x] = median(preds[L==i]) //Centroid of the prediction for each user.
	End For

Step6:	//Assign an optimal value for each user.
	M = 1000000000/2 //The center of the values.
	FI ← 0 //Index of the farthest point from M in O.
	FD ← 0 //Maximum distance between M and the optimal points.

	While len(O) > 0:
	For i = 1 to len(O):
	If Absolute (M - O[i]) > FD:
	FD ← Absolute (M - O[i]) //Update the farthest distance.
	FI ← i //Update the index of the farthest optimal point.
	End If
	End For
	CD ← 1000000000 //Smallest distance between O[FI] and centroids.
	CI ← 1000000000 //Index of the nearest centroid to O[FI].
	For i = 1 to len(C):
	If Absolute (C-O[FI]) < CD:
	CD ← Absolute (C-O[FI])
	CI ← i
	End If
	End For
	OL[L == CI] ← O[FI] //Assign the optimal value to the user with the nearest centroid.
	Delete O[FI] //Delete the assigned optimal point from list.
	Delete C[CI] //Delete the centroid of the user.
	End While

Step7:	//Train the neural network using the assigned optimal values.
	nn.train(F, OL, epochs=50)
	T ← T+1

Step8:	If T<1000: Go to Step3 End If
Step9:	Return nn END

The optimal distribution of the value to be assigned to the individuals in the training dataset is calculated in the space defined for these values, which is set to 1,000,000,000. This value is selected according to the ability of using a maximum of 4,294,967,295, i.e. integer with 32-bit size, as a seed for the

random generator. This allows to maximize the distance between two consequential optimal points, S in Algorithm (3.3), while allowing more flexibility for the neural network, in runtime, to output larger values. Moreover, as the output of the ReLU activation function is always positive, a deviation from the zero value D is used to ensure that the neural network does not learn to divert toward negative values, which can reduce the uniqueness of the produced values. Figure (3.4) shows the optimal distribution of a sample dataset with face images of 20 individuals.

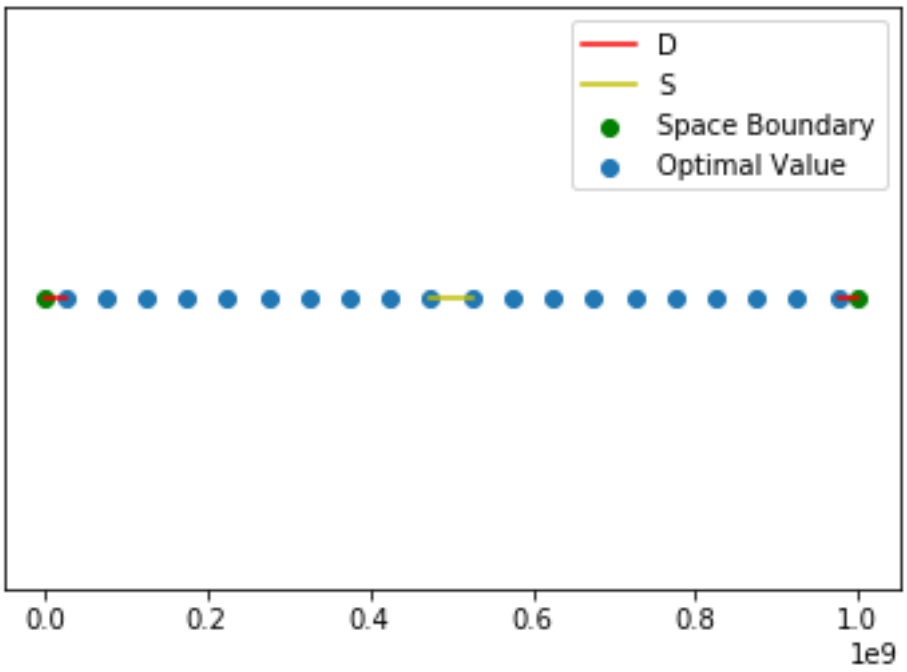


Figure (3.4): Optimal distribution of the values to be assigned to individuals in a sample training dataset.

Moreover, as shown in Step6 in Algorithm (3.3), the assignment of the values is initiated from the farthest point from the center of the optimal values. As soon as this value is recognized, it is assigned to the user with the nearest centroid. This assignment ensures a relative expansion of the values assigned to the user, i.e. the largest optimal value is assigned to the user with the largest centroid and vice versa. Hence, the features detected by the neural network are

maintained, as the relative positions of the predicted values are maintained while being expanded.

After training the neural network for 50 epochs, the predictions are recollected and the distribution of the optimal values is updated to include any new features detected by the neural network in further training. Eventually, as the values used for the training are equal for each user and different for different users, the neural network recognizes the features that can be used to produce these values. However, the values used for the training are selected based on the predictions of the neural network to avoid imposing values that cannot be interpreted by the neural network.

D. Secret Key Generation

As illustrated in Section (2.3), the user's secret key used to produce the digital signature is *x*, which is a random number in the interval [0, q]. To produce this value, the random seed retrieved from the face image using the artificial neural network is used to produce a series of values that are used to produce the value of x with the size defined by the administrator, as shown in Algorithm (3.4).

Algorithm (3.4): Private component generation of the DSA'a secret key.

Input: **Output:**	Random seed, Size of x. The private component of the secret key x.
Step1:	S ← Read the random seed. RG ← Initiate a random generator using the random seed S. L ← Read the length of the required value in bytes.
Step2:	//generate a random number with size L based on the random seed S. B ← [] //Initiate an empty array to store the random values. For i =1 to L: B ← [B, RG.generate_random_number(0,255)] //Append a new single-byte random

	value. End For
Step3:	x ← Bytes_to_integer(B) //Convert the bytes into a single integer value. Return x

As the random generator is used to produce the bytes of the x value, the use of the same random seed ensures the production of the same value. As the seed values are extracted from the face images and may vary for each user, it is important to ensure that the generated secret key for a certain user does not conflict with another user's in the database.

E. Secret Key Selection and Validation

To avoid any conflict between the secret key generated for the user being registered into the system and the key of any other previous user, the generated key must be validated. However, as the secret keys of the previous users are not stored in the database, the validation procedure relies on the other users' public keys, as shown in Algorithm (3.5).

Algorithm (3.5): Key selection and validation algorithm.

Input: **Output:**	Faces descriptors, Size of x. The user's secret key.
Step1:	FD ← Read faces descriptors. L ← Read the length of the required value in bytes.

Step2:	//Generate a random seed for every face descriptor.
	S ← nn.predict(FD) //Predict random seeds using the trained neural network from Algorithm (3.3)
	F, V = Calculate the frequency of each value in S. //F holds the frequency of each value in V.

Step3:	//Find the most frequent seed value to generate and validate the secret key. VK ← False. //Validity of the generated key. While VK is False:

```
        VK ← True
        mv ←  max(F). //Find the highest frequency.
        mvi ← argwhere(F == mv). //Index of the highest frequency.
        x ← Generate the private component of the key using
              Algorithm (3.4) and V[mvi] with size L.
        sk ← Generate user's secret key using DSA algorithm
        rs ← Generate a random string.
        srs ← Sign rs using sk as in DSA algorithm
        For each public key pk in the database:
              If srs is valid with pk using DSA algorithm
                    VK ← False. //Key is invalid
                    Break. //Quit verification process.
              End If
        End For
        Delete F[mvi]
        Delete V[mvi]
End While
If VK is True:
        Return sk
Else:
        Return None
End If ,END
```

As shown in Algorithm (3.5), if the signed message is validated using another user's public key, the key generated for the new user is considered invalid and the next most frequent value is used to generate another key. Moreover, if all the values are used to generate the private keys and none of them is found valid, a *None* value is returned, so that another set of face images can be collected to repeat the key generation and selection procedure.

F. Model Descriptor Selection

As the seeds for the random generator are extracted from the descriptor of the user's face image and as such descriptor must be stored in the database for authentication purposes, storing a descriptor that can be used to generate the

random seed can pose a security threat to the proposed method. Thus, to avoid such a threat, the descriptors that produce the selected seed values are eliminated from the collected set. Then, the descriptor that generates the least frequent seeds is stored in the database, to be used for authentication purposes. Thus, even if the database of the proposed system is compromised, the intruder cannot retrieve the private keys of the users, hence, cannot digitally sign any document. This approach also shows the importance of producing balanced robustness for the generated keys, so that more options are available for the selected descriptor, as well as the selection of the keys in the previous step. Perfectly robust values mean that the extraction of the private key can be achieved using any of the descriptors generated for the user's face images.

3.2.2.Digital Signature Generation

In this section, the procedure of producing a digital signature for a document using the proposed system is illustrated. An overview of the document signing procedure by registered users is shown in Algorithm (3.6).

Algorithm (3.6): Overview Of The Document signing procedure.

Input:	User's login credentials; Live video frames, Digital document.
Output:	Digital signature.

Step1:	//Authenticate the user using username and password.
	D ← Read the digital document.
	C ← 0
	While C < 5:
	UU, UP ← Input User's username and password.
	HUP ← Calculate the hash of UP using SHA-256 algorithm
	HP ← Retrieve the hash of the password for user UU from database.
	If HUP == HP:
	Go to Step2
	End If
	Go to Step5 //Terminate the procedure.

Step2:	C ← 0
	VF ← [(30×160×160)] //Empty array with (30×160×160) dimensions.
	While C<30:
	F ← Capture video frame.
	x, w, y, h ← Detect face image in F. //using MTCNN
	If x, w, y, h are not *null*:
	F ← F[x:x+w, y:y+h] //Extract the region that has the face image.
	F ← Resize F to 160×160. //using Bilinear interpolation
	VF[C] ← F
	C ← C+1 //Increase the counter by one.
	End If
	End While

Step3:	If liveness is detected in VF: //algorithm (3.2.1.B) FD ← Generate faces descriptors for VF. //using facenet If user is authenticated using FD: //algorithm (3.2.2.A) Go to Step4 Else: Go to Step5 //Terminate the procedure. End If End If
Step4:	sk ← Generate DSA's secret key using FD. //algorithm (3.2.2.B) DS ← Generate digital signature for document D with sk using DSA algorithm , END

A. User Authentication with Face Images

After the username and password of the user are validated using the hash of the password, similar to the administrator authentication procedure shown in Section (3.2.1.A), the live face images of the user are used to authenticate the user to improve the security of the system. Instead of using the Euclidean distance between each descriptor in the collected set of face images and the

model descriptor of the user, the neural network shown in Figure (3.5) is implemented and trained to predict the appropriate authentication decision. The model descriptor is retrieved from the database of the proposed system based on a username and password entered by the user to use it in the authentication process. Then, a decision is predicted for each face image, i.e. a total of 30 decisions are collected and used to authenticate the user. Therefore, the formula shown in Equation (3.2) is used to summarize these 30 values into a single value. This value produced by the formula is compared to a threshold of 0.5, where users with similarity measures equal to or greater than the threshold value are granted access to the system. The neural network is trained using a set of positive and negative triples generated from a dataset that contains face images of multiple individuals.

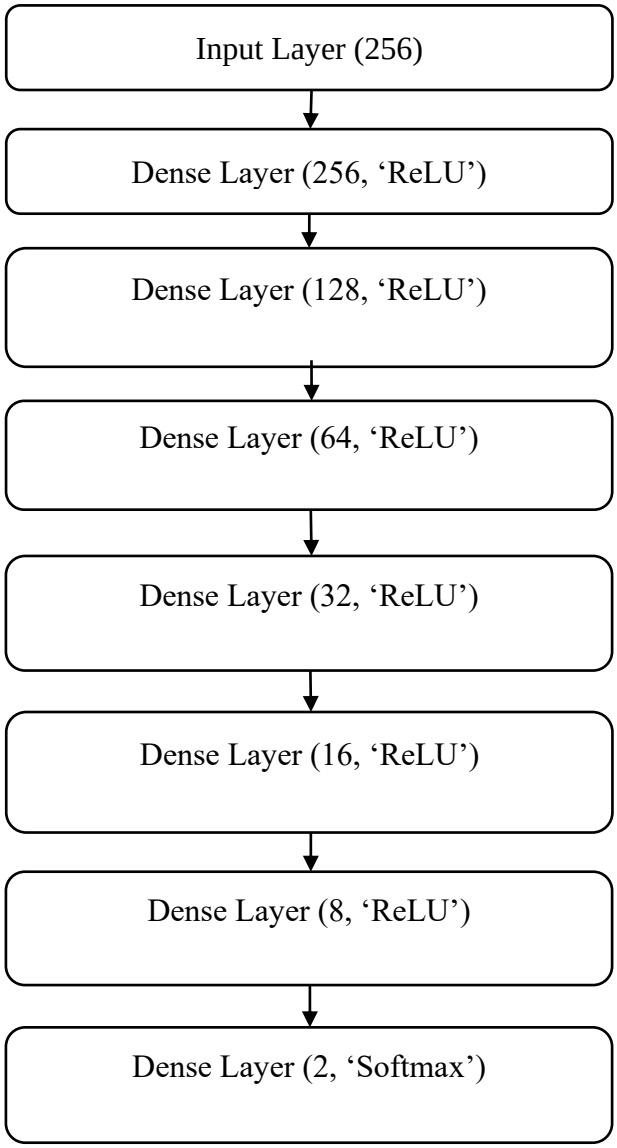


Figure (3.5): The implemented neural network for users authentication.

Each triple consists of a template descriptor, a positive sample collected from the same individual, and a negative sample collected from a different individual. The positive pair is labeled with [0, 1] while the negative pair is labeled as [1, 0], so that each neuron in the output layer represents a decision from the network. In this approach, the overall authenticity measure (m) can be calculated as the average of the number of face images that have higher output in the second neuron of the predictions from the output layer, as shown in Equation (3.2).

$$m = \frac{\sum_{i=1}^{30} argmax(prediction_i)}{30} \quad (3.1)$$

The neural network can provide more accurate predictions than the use of the Euclidean distance, as shown in Section (4.3).

B. Private Key Generation and Validation

Unlike the validation process in the registration procedure, the private key based on the numbers generated from the random seeds extracted from the face image is validated against the public key of the authenticated user. This step is to ensure that the signature can be validated using the user's public key in remote terminals, as perfect robustness cannot be guaranteed as well as the selection of the most frequent values. Thus, Algorithm (3.7) is used to validate the generated private key.

Algorithm (3.7): Authenticated user's private key validation algorithm.

Input:	Faces descriptors, Size of x.
Output:	The user's secret key.
Step1:	FD ← Read faces descriptors. L ← Read the length of the required value in bytes.

Step2:	//Generate a random seed for every face descriptor.
	S ← nn.predict(FD) //Predict random seeds using the trained neural network from Algorithm (3.3)
	F, V = Calculate the frequency of each value in S. //F holds the frequency of each value in V.

Step3:	//Find the most frequent seed value to generate and validate the secret key. VK ← False. //Validity of the generated key. While VK is False: mv ← max(F). //Find the highest frequency. mvi ← argwhere(F == mv). //Index of the highest frequency. x ← Generate the private component of the key using Algorithm (3.4) and V[mvi] with size L. sk ← Generate user's secret key using DSA Algorithm. rs ← Generate a random string.

	srs ← Sign rs using sk as in Algorithm (2.3).
	pk ← Retrieve the user's public key from the database.
	If srs is valid with pk using DSA Algorithm
	VK ← True. //Key is valid.
	End If
	Delete F[mvi]
	Delete V[mvi]
	End While
	If VK is True:
	Return sk
	Else:
	Return None
	End If , END

Similar to the validation procedure in the registration step, a *None* value is returned by the key validation algorithm when no valid keys are found in the current set of face descriptors. This value can be used to collect a new set of face images and create their descriptors to search for the valid private key. Thus, extremely low robustness of the calculated random seeds can significantly increase the time required to find the valid private key in the digital signature step.

3.2.3.Signature Verification

Upon arrival, a digitally signed document contains the original document and the digital signature produced by the sender to approve the sent document. To verify the signature, the user is first required to authenticate into the system using a username and password, as well as face authentication to improve the security measures of the proposed system, as shown in Algorithm (3.8).

Algorithm (3.8): signed digital documents verification.

Input:	User's login credentials; Live video frames, Signed digital document.
Output:	Validated

Step1:	//Authenticate the user using username and password.
	D, S ← Read the digital document and the corresponding signature.
	C ← 0
	While C < 3:
	UU, UP ← Input User's username and password.
	HUP ← Calculate the hash of UP using SHA-256 Algorithm.
	HP ← Retrieve the hash of the password for user UU from database.
	If HUP == HP:
	Go to Step2
	End If
	Go to Step5 //Terminate the procedure.

Step2:	C ← 0
	VF ← [(30×160×160)] //Empty array with (30×160×160) dimensions.
	While C < 30:
	F ← Capture video frame.
	x, w, y, h ← Detect face image in F. //using MTCNN
	If x, w, y, h are not *null*:
	F ← F[x:x+w, y:y+h] //Extract the region that has the face image.
	F ← Resize F to 160×160. //using Bilinear interpolation
	VF[C] ← F
	C ← C+1 //Increase the counter by one.
	End If
	End While

Step3:	If liveness is detected in VF: //Section(3.2.1.B) FD ← Generate faces descriptors for VF. //using Facenet If user is authenticated using FD: //Section (3.2.2.A) Go to Step4 Else:

	Go to Step5 //Terminate the procedure. End If End If
Step4:	pk ← Retrieve sender's DSA public key from the database. Verify the signature of D using pk and S as shown in Algorithm (2.3). END

Chapter Four

Chapter Four

Experimental Results and Performance Evaluation

4.1 Introduction

The performance of the methods proposed in this thesis is evaluated by conducting a set of experiments. These experiments and their results are described in this chapter. The performance of each method is evaluated individually first, then, the overall performance of the system is evaluated. As the liveness detection and face authentication are being used in different parts of the system, the accuracy of these methods is evaluated first. Then, the performance of each stage is evaluated including all the required steps. All experiments are conducted using a computer running Windows operating system with an Intel® Core™ i7-7700HQ processor with a memory of 16GB. The computer also contains an Nvidia GTX1080Ti with 8GB of memory that is used to accelerate the neural networks' computations. All methods are implemented and evaluated using the Python programming language.

Two datasets are used for performance evaluation. One dataset is the colored FERET dataset[55], which contains 269 individuals face images. Each individual has a minimum of 11 images for different posing angles, while some of the users have multiple sets producing a total of 3528 images in the dataset. The other dataset is collected from volunteers using a live video capturing device, i.e. a camera. A total of 33 videos is collected from the 33 volunteers at a frame rate of 30 frames per second and a resolution of 640×480 pixels. for each video, the volunteer is required to move his head in different angles, upward, downward , and sideways to simulate a real possible positioning of the users of the proposed system. The durations of the videos vary from 12.6S to 92.5S with an average duration of 41.83S. These videos are used to replicate the use of the proposed method in real-time in order to evaluate its performance.

4.2 Robustness and Uniqueness Performance Measures

According to the novelty of the proposed method, a benchmark is used to evaluate the proposed method and to be used by any future researchers to evaluate their methods and compare their performances to the method proposed. The benchmark is defined according to the requirement of the random seed. The first performance measure is the robustness of the values produced by the neural network. Per each user *u*, the robustness of the predictions is equal to the frequency of the most frequent value *f* in the predictions divided by the number of predictions provided by the neural network *p*, which is equal to the number of face images collected from that user, as :

$$Robustness_u = \frac{|f_u|}{|p_u|} \tag{4.1}$$

The overall robustness of the system is calculated as the average of the robustness for all the users in the evaluation dataset. The other important performance measure is the uniqueness of the values produced for the user, where the value produced for a certain user is required to be unique in the predictions. Hence, the uniqueness of the values produced for the user *u* is calculated as:

$$Uniqueness_u = \frac{1}{\sum_{i=1}^{U} \begin{cases} 1 & f_i = f_u \\ 0 & f_i \neq f_u \end{cases}} \tag{4.2}$$

As the formula shows, the robustness for a unique value is equal to 100%, while if this value is found as the most frequent value for another individual, the robustness becomes 50%. The overall robustness of the system is also calculated as the average of robustness values of all users. Moreover, for accurate evaluation, the images used for the evaluation phase must be for individuals that

are not included in the training phase. Instead of splitting the face images into training and testing sets, the individuals are split, then the images of the users in the training set are used for training while those for individuals in the training sets are used for the evaluation. This split ensures that the face images included in the evaluation have never been recognized in the training of the neural network. Hence, the produced evaluation measures are accurate and unbiased .

4.3 Results of Face Detection and Recognition

As shown in Section (2.7.1), the MTCNN has relatively better performance, compared to the other state-of-the-art techniques in the literature. Thus, this technique is used for face detection in the proposed method. The performance of this method is evaluated using the real-time videos collected from the volunteers. As each frame contains a face image, which can be in different positionings, the ratio between the number of frames detected to have face images by the MTCNN to the total number of frames in the video is illustrated for each user in Table (4.1).

Table (4.1): Performance measures of the MTCNN face detection using the collected real-time dataset.

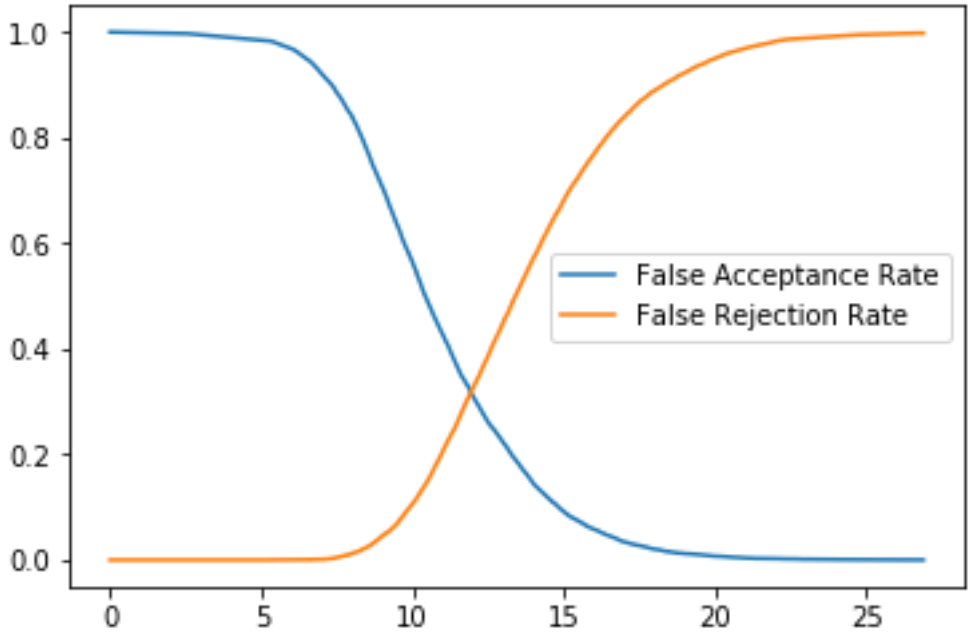
Individual	Frames Count	Detected Faces	Recall	Detection Time (Seconds)
1	475	440	92.63%	0.0692
2	775	562	72.52%	0.0701
3	925	561	60.65%	0.0570
4	266	256	96.24%	0.0573
5	473	387	81.82%	0.0575
6	725	364	50.21%	0.0508

Individual	Frames Count	Detected Faces	Recall	Detection Time (Seconds)
7	818	367	44.87%	0.0515
8	684	301	44.01%	0.0527
9	408	325	79.66%	0.0568
10	503	440	87.48%	0.0665
11	449	280	62.36%	0.0619
12	381	352	92.39%	0.0630
13	383	371	96.87%	0.0618
14	283	260	91.87%	0.0585
15	388	356	91.75%	0.0581
16	363	348	95.87%	0.0585
17	380	339	89.21%	0.0590
18	519	297	57.23%	0.0565
19	387	304	78.55%	0.0569
20	358	328	91.62%	0.0611
21	539	406	75.32%	0.0568
22	352	350	99.43%	0.0577
23	397	369	92.95%	0.0569
24	375	375	100.00%	0.0577
25	364	337	92.58%	0.0562

Individual	Frames Count	Detected Faces	Recall	Detection Time (Seconds)
26	283	276	97.53%	0.0577
27	509	494	97.05%	0.0572
28	196	179	91.33%	0.0575
29	126	126	100.00%	0.0582
30	218	184	84.40%	0.0573
31	175	170	97.14%	0.0639
32	170	150	88.24%	0.0650
33	156	136	87.18%	0.0600

Additionally, the Facenet neural network has been able to outperform other methods for similar facial features generation from face images, as shown in section (2.7.5). The existing techniques rely on measuring the Euclidean distances between the vectors generated by this neural network for each face to measure their similarity. The method proposed in this thesis uses a neural network to measure the similarity between the vectors, which allows better decision as neural networks provide more nonlinearity. The proposed neural network, shown in Figure (3.5), is implemented and trained using the FERET dataset using triplet loss as illustrated in section (2.7.5). To illustrate the enhancement in the performance of the system by using the proposed method, the first face image of each of the collected videos is used as a template, whereas the face images from the remaining frames are used as the positive, i.e. legitimate. Inputs and face images are collected from other videos to represent false, or intrusion, attempts. The use of the Euclidean distance has achieved an

authentication accuracy of 68.20% at Equal Error Rate (EER) with a threshold value of 11.93, as shown in Figure (4.1).



Figure(4.1): ROC curve of the face authentication using the Euclidean distance.

In contrast, the proposed neural-network-based authentication method produces binary decisions instead of linear similarity measures, which eliminates the need for a threshold value to compare it with. The proposed method has been able to achieve predictions' accuracy of 93.92%, according to the confusion matrix shown in Table 4.2, whereas the quality of the predictions is illustrated in Table 4.3.

Table (4.2): Confusion matrix of the predictions of the face authentication neural network.

		Predicted	
		Fake	**Real**
Actual	**Fake**	10138	652
	Real	660	10130

Table(4.3): Summary of the performance measures of the proposed face authentication neural network.

	Precision	Recall	Fa-score	Support
Fake	0.9398	0.9396	0.9392	10790
Real	0.9395	0.9388	0.9392	10790
Avg / Total	0.9392	0.9392	0.9392	21580

4.4 Results of Liveness Detection

The proposed liveness detection method is evaluated using the real-time videos collected from the volunteers. However, according to the need of fake face images, i.e. spoofing face images, these videos are played back using a smartphone in front of the computer camera, as shown in Figure (4.2). The videos collected from this scenario are used as the negative samples to train the neural network. Samples frames of an actual and fake videos are shown in Figure (4.3).

Figure (4.2): Spoofing attack videos collection setup.

Figure (4.3): Sample frames of fake and real videos; Left: Real video frame; Right: Fake video frame.

Both real and spoof videos are combined in a single dataset, which is randomly split into 80% for training and 20% for testing datasets. The proposed neural network is implemented and trained for 100 epochs using 300 batch size. To avoid any confusion to the neural network, the face images extracted from the frames using boundaries defined by the MTCNN are only resized to 160×160 pixels, without manipulating the color intensity values. By the end of the training, the proposed liveness detection method has been able to achieve 99.96% prediction accuracy, as shown in the conduction matrix in Table (4.4). The precision, recall, and f1-score of the liveness detection predictions are shown in Table (4.5).

Table (4.4): Confusion matrix of the predictions of the liveness detection neural network.

		Predicted	
		Fake	**Real**
Actual	**Fake**	2155	2
	Real	1	5128

Table(4.5): Summary of the performance measures of the proposed liveness detection neural network.

	Precision	**Recall**	**F1-score**	**Support**
Fake	0.9995	0.9991	0.9993	2157
Real	0.9996	0.9998	0.9997	5129
Avg / Total	0.9996	0.9996	0.9996	7286

The performance of the trained neural network is evaluated using all the videos in the dataset by following the approach presented in Algorithm (3.1). The minimum accuracy for each 30-frame batch collected from the real and fake videos per user is shown in Table (4.6). These results show that the minimum accuracy for all the users' batches is 93.33%. Hence, the threshold to detect spoofing attacks can be set to 90% for the proposed method.

Table (4.6): Minimum accuracy for real and fake videos liveness detection.

Individual	Real Video Accuracy (%)	Fake Video Accuracy (%)
1	100	100
2	100	100
3	100	100
4	97	100
5	93	97
6	100	100
7	100	100
8	100	100
9	97	100
10	100	100
11	100	93
12	100	100
13	100	100
14	100	100
15	93	100
16	100	100
17	100	100
18	100	100
19	100	100
20	100	93
21	100	100
22	100	100
23	93	97
24	100	100
25	100	100
26	100	100

Individual	Real Video Accuracy (%)	Fake Video Accuracy (%)
27	100	100
28	100	100
29	100	100
30	100	100
31	100	97
32	97	100
33	100	100

4.5 Results of Random Seeds Generation

To generate random seeds for private and public keys of the users based on their facial features, the neural network shown in Figure (3.3) is implemented and trained using the approach shown in Algorithm (3.3). The quality of the predictions provided by this neural network is measured by their uniqueness and robustness for each user. The training of the neural network is conducted using 80% of the 269 individuals in the FERET dataset, i.e. 215 individuals. The performance of the random seed generator is evaluated using the remaining 54 individuals, whereas 5 of these individuals have less than two face images detected by the MTCNN neural network, which are eliminated from the evaluation. Then, the performance is evaluated using the videos collected from the volunteers to replicate the use of the proposed method in real-time. The training is conducted for 1000 iterations with 50 training epochs per each iteration. Figure (4.4) illustrates the distribution of the centroids of the predictions of the neural network, the optimal distribution of the assigned values, and the normalized centroid values. Although these values are scalars, different types of values are distributed at different levels on the y-axis for better illustration.

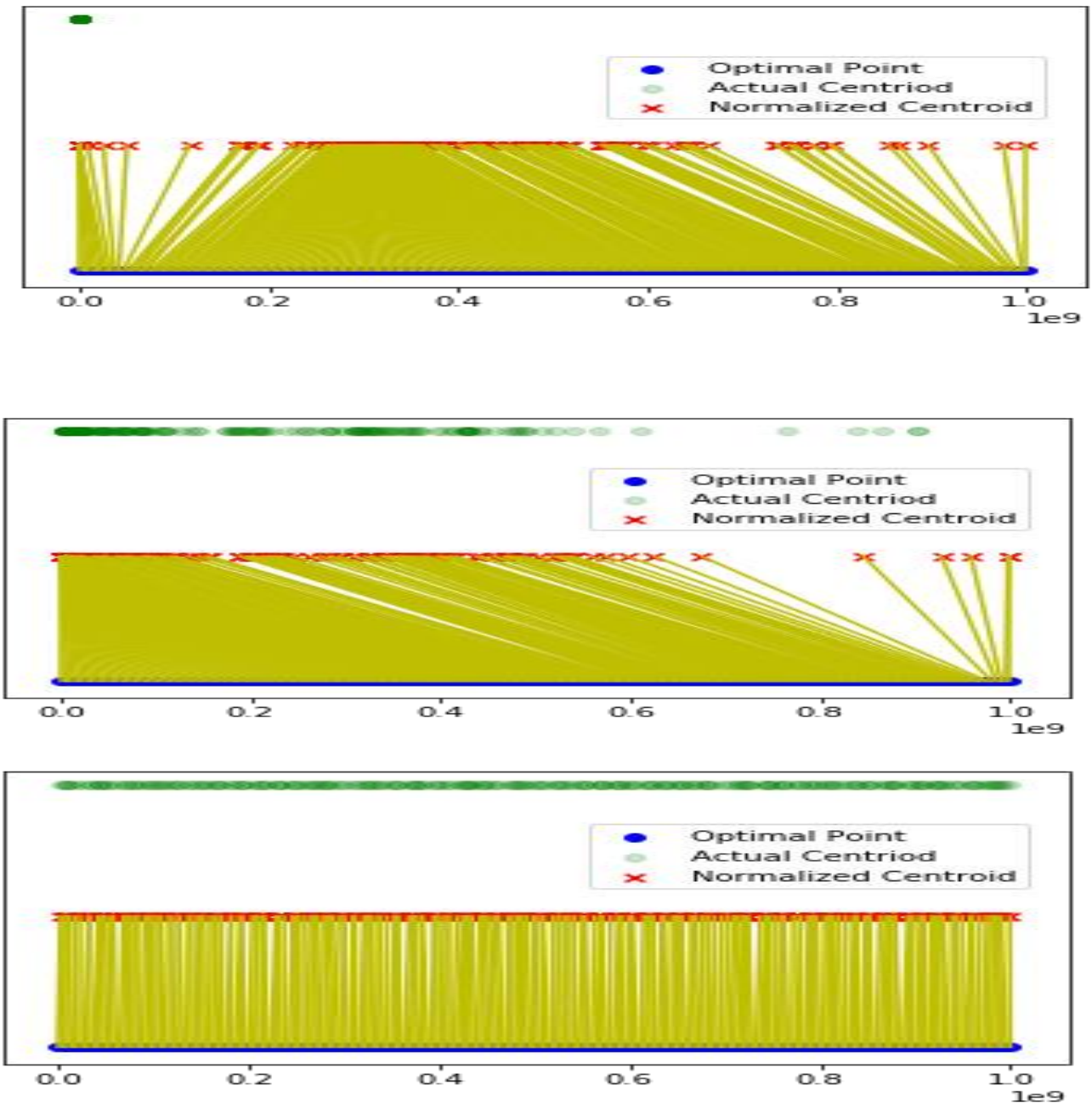


Figure (4.4): Distribution of the prediction's centroid, normalized centroids and optimal distribution; above: 1st iteration; Center: 500th iteration; under: 1000th iteration.

Table (4.7) illustrates the performance of the random seed generation for each of the users in the testing FERET dataset . The robustness represents the ratio between the batches collected from the video that produce the same seed values to the total number of batches in the video, where each batch contains 30 frames, as shown in Algorithm (3.1). The uniqueness, on the other hand, represents the ratio of duplicate random seeds to the total number of individuals in the dataset.

Table (4.7): Performance of the random seed generation neural network using the test split of the FERET dataset.

Individual	Identical	Total	Robustness	Uniqueness	Time(*m*S)
1	4	9	33.33%	100%	6.979
2	8	15	46.67%	100%	2.019
3	7	9	66.67%	100%	1.995
4	5	13	30.77%	100%	2.99
5	8	10	70.00%	100%	1.996
6	3	5	40.00%	100%	1.993
7	8	9	77.78%	100%	2.997
8	3	8	25.00%	100%	1.992
9	5	7	57.14%	100%	2.001
10	6	9	55.56%	100%	2.984
11	3	8	25.00%	100%	2.009
12	8	14	50.00%	100%	1.981
13	5	8	50.00%	100%	1.995

Individual	Identical	Total	Robustness	Uniqueness	Time(*m*S)
14	6	7	71.43%	100%	2.992
15	2	4	25.00%	100%	1.997
16	4	8	37.50%	100%	2.005
17	19	39	46.15%	100%	4.977
18	3	6	33.33%	100%	1.994
19	5	9	44.44%	100%	1.994
20	6	9	55.56%	100%	1.995
21	7	8	75.00%	100%	2.99
22	5	7	57.14%	100%	1.997
23	6	8	62.50%	100%	2.01
24	3	4	50.00%	100%	2.983
25	5	10	40.00%	100%	3.002
26	4	8	37.50%	100%	2.979
27	6	14	35.71%	100%	2.993
28	5	10	40.00%	100%	1.994
29	2	4	25.00%	100%	2.013
30	13	19	63.16%	100%	2.974
31	4	9	33.33%	100%	2.996
32	5	11	36.36%	100%	1.991
33	12	19	57.89%	100%	1.995

Individual	Identical	Total	Robustness	Uniqueness	Time(*m*S)
34	5	9	44.44%	100%	2.027
35	5	9	44.44%	100%	0.966
36	4	10	30.00%	100%	2.007
37	2	4	25.00%	100%	2
38	2	5	20.00%	100%	2.007
39	6	8	62.50%	100%	1.981
40	9	16	50.00%	100%	1.994
41	2	4	25.00%	100%	1.994
42	6	10	50.00%	100%	2.018
43	4	9	33.33%	100%	1
44	4	9	33.33%	100%	1.975
45	5	8	50.00%	100%	1.989
46	8	10	70.00%	100%	2.003
47	9	10	80.00%	100%	1.986
48	6	9	55.56%	100%	0.997
49	4	9	33.33%	100%	1.995
Average	**5.6327**	**9.71**	**46.16%**	**100%**	**2.32**

The results of this evaluation show that even with a low number of face images available for each individual, compared to real-life scenarios when 30 face images can be collected per each second, the proposed neural network has

been able to successfully produce the required random seeds. However, face images collected from video frames can be less descriptive than standstill images according to the movement of the user, which can produce some blurriness to images. Thus, the videos collected from the volunteers are used for the evaluation and the performance of the proposed random seed generation neural network as shown in Table (4.8).

Table (4.8): Performance of the random seed generation neural network using the videos collected from the volunteers.

Individual	Identical	Total	Robustness	Uniqueness	Time(mS)
1	68	440	15.23%	100%	36.928
2	96	562	16.90%	100%	37.872
3	99	561	17.47%	100%	35.903
4	44	256	16.80%	100%	12.967
5	72	387	18.35%	100%	18.918
6	68	364	18.41%	100%	17.954
7	62	367	16.62%	100%	16.954
8	64	301	20.93%	100%	15.009
9	55	325	16.62%	100%	18.901
10	78	440	17.50%	100%	23.969
11	55	280	19.29%	100%	14.002
12	74	352	20.74%	100%	16.955
13	65	371	17.25%	100%	19.946
14	47	260	17.69%	100%	13.963

Individual	Identical	Total	Robustness	Uniqueness	Time(mS)
15	67	356	18.54%	100%	19.946
16	82	348	23.28%	100%	22.938
17	53	339	15.34%	100%	21.942
18	60	297	19.87%	100%	16.954
19	60	304	19.41%	100%	16.954
20	75	328	22.56%	100%	18.95
21	91	406	22.17%	100%	26.929
22	68	350	19.14%	100%	17.951
23	58	369	15.45%	100%	18.982
24	124	375	32.80%	100%	18.918
25	137	337	40.36%	100%	17.951
26	57	276	20.29%	100%	14.962
27	83	494	16.60%	100%	27.937
28	25	179	13.41%	100%	10.958
29	32	126	24.60%	100%	5.988
30	32	184	16.85%	100%	10.997
31	31	170	17.65%	100%	7.974
32	32	150	20.67%	100%	8.977
33	31	136	22.06%	100%	7.979
Average	**65**	**326.97**	**19.72%**	**100%**	**18.619**

These results show a reduction in the average robustness, which is a result of the less descriptive face images that can produce larger variation in the descriptors generated by the FaceNet neural network. However, in both cases the proposed method has been able to maintain 100% uniqueness, i.e. no random seed value is found duplicate among different users, and applicable rate of robustness. The 19.72% average robustness indicates that about 6 of the frames in the 30-frame batch collected from the user are expected to produce the seed value that can be used to calculate the private key of the user.

4.6 Results of Keys Generation and Validation

As the keys are generated based on the seeds predicted by the neural network, based on the facial features retrieved from the FaceNet neural network, the computations of the private and public keys are not guaranteed to match the ones assigned to the user during registration. Thus, it is important to validate the keys using the user's public key, which is stored in the database, as shown in Algorithms (3.5) and (3.7). For user registration, the private and public keys extracted from the first 30-frame batch from the video. If the generated private key conflicts with another user's, based on their public keys, the next batch is selected and so on until a valid private key is generated. For document signing, the private key is validated against the same user's public key in the database, using frame batches positioned behind the batch selected for the registration. For each user, the number of batches required to calculate valid keys for registration and documents signing is measured, as well as the time required to compute these keys under these conditions. The results of this experiment are summarized in Table (4.9).

Table (4.9): The number of batches and time required to generate the private and public keys during registration and signing phases.

No	Registration		Signing	
Individual	Number of Batches	Time(S)	Number of Batches	Time(S)
1	1	1.077	4	4.192
2	1	1.078	6	6.331
3	1	1.081	3	3.138
4	1	1.078	7	7.338
5	1	1.092	6	6.358
6	1	1.091	5	5.316
7	1	1.086	3	3.175
8	1	1.077	2	2.106
9	1	1.071	6	6.338
10	1	1.076	3	3.190
11	1	1.079	6	6.358
12	1	1.094	4	4.182
13	1	1.071	3	3.165
14	1	1.079	6	6.361
15	1	1.093	5	5.316
16	1	1.072	1	1.055
17	1	1.090	2	2.112
18	1	1.090	1	1.048
19	1	1.086	6	6.384
20	1	1.092	2	2.113

No.	Registration		Signing	
Individual	Number of Batches	Time(S)	Individual	Number of Batches
21	1	1.078	5	5.257
22	1	1.082	5	5.242
23	1	1.082	2	2.120
24	1	1.083	4	4.209
25	1	1.086	3	3.136
26	1	1.083	4	4.194
27	1	1.083	2	2.106
28	1	1.081	6	6.325
29	1	1.078	6	6.294
30	1	1.074	3	3.139
31	1	1.077	3	3.182
32	1	1.082	5	5.294
33	1	1.077	5	5.322
Average	**1**	**1.082**	**4.061**	**4.285**

The results show that a single batch has always been successful in producing valid keys upon registration according to the high uniqueness of the random generator. As the keys generated upon registration are validated against the other users' keys, the high uniqueness indicates that no duplicates exist in the predictions, hence, no conflicts are found in the generated keys. However, according to the lower uniqueness of the random seed generator, multiple batches are required to find a valid secret key for the user. As the batch contains 30 face images collected from 30 frames, the time required to produce a valid

private key to sign the documents in seconds is approximately equal to the number of batches. Hence, as shown in Figure (4.5), the longest interval required to produce a digital signature is approximately seven seconds, as seven batches are required to reproduce the private key of the user. However, only a single user has required such large number of frame batches, whereas the private keys of the remaining users have been reproduced using less batches. The private keys of two of the individuals in the collected real-time videos have been recognized using a single batch, i.e. the digital signatures are produced in approximately one second.

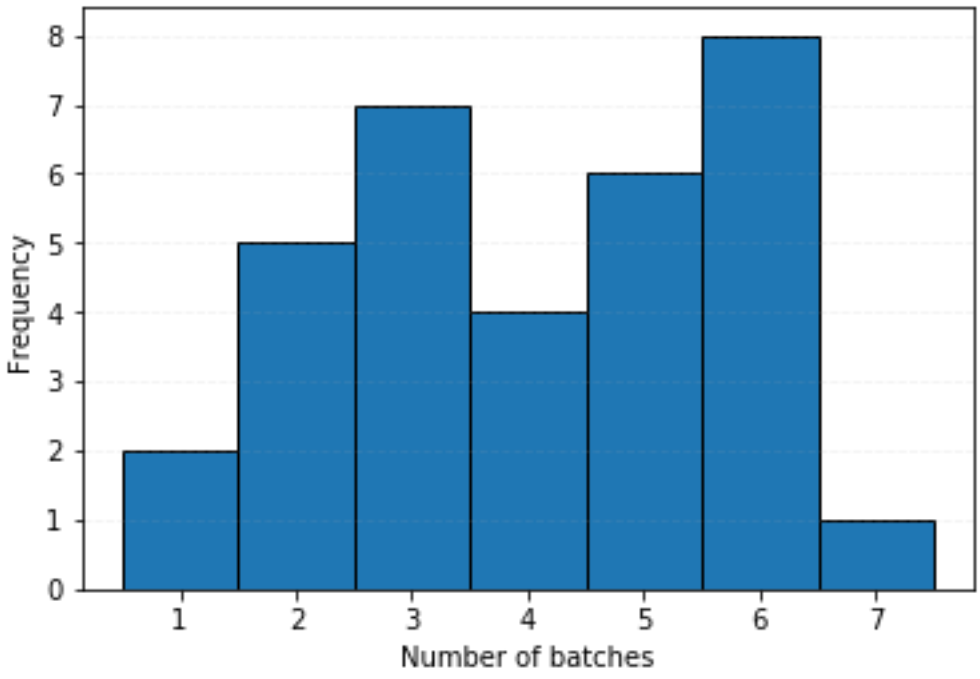


Figure (4.5): Histogram of the number of batches required to produce the users' private keys.

4.7 Attacking the Generated Keys

The liability of the DS system is defined by the security of the secret keys, as gaining access to these keys allows attackers to produce false signatures. However, the use of the proposed method can improve the security of the signatures against the attacks described in Section (2.11) in table 4.10.

Table (4.10): Summary of attack resistance stages in the proposed method.

Attack	Prevention Stage	Description
Brute Force Attack	DSA Keys	The use of DSA keys to sign the documents allows the proposed method to resist brute force attack, according to the extensive processing required to compute these keys.
Dictionary Attack	Face Authentication	The use of face authentication instead of passwords eliminates the risk of applying a dictionary attack.
Phishing Attack	Liveness Detection	A replicate of a legitimate user's face can be detected at the liveness detection stage.
Shoulder Surfing Attack	Face Authentication	The use of face authentication instead of passwords denies shoulder surfing attacks.
Guessing Attack	Face Authentication	The enormous possible face shapes eliminate the probability of guessing the legitimate user's face image.

Chapter Five

Chapter Five

Conclusion and Future Work

5.1 Conclusion

In this thesis, a novel method is introduced to generate user's secret keys from their facial features directly without the need to store them. The main features of the proposed method are summarized as follows:

- The secret keys are generated based on a random seed produced by a neural network, which uses the facial feature extracted from the users as its inputs. To train this neural network to produce values that are similar for the same user and different from other users', a semi-supervised training approach is used. This approach relies on random seed values produced by the neural network to assign optimal values for the users.
- The use of such approach allows assigning a random seed value that is most appropriate for the user according to the features detected by the neural network.
- As the random seed generated by the neural network cannot be guaranteed to be the same every time extracted from facial features, the generated secret keys are validated at both the user registration and documents signing stages.
- The proposed method uses a liveness detection stage to immunize the system against any spoofing attacks and ensure the existence of the legitimate user in real-time during the production of the signature.
- Face images are represented using three-dimensional arrays, i.e. including the color channels or information, in order to increase the accuracy of the system.
- The experiments conducted to evaluate the performance of the proposed method using facial features collected from real-life volunteers has shown

that the random seed generation neural network has 19.72% robustness and 100%uniqueness. The results also show that the proposed method has been able to find a unique secret key for each of the 33 volunteers using an average of 1.082 batches of face images, each batch contains 30 images collected in real-time in one second, and to successfully produce a signature using an average of 4.061 batches.

5.2 Suggestions for Future Work

The following improvements can be investigated in future work to improve the performance of the proposed method:

1. Using other biometric templates such as fingerprint and iris to authenticate users and extract secret keys. Despite the need of more expensive equipment to capture fingerprint or iris templates, the higher robustness of these templates compared to the use of faces can improve the robustness of the random seeds generated by the neural network.
2. Using the proposed method with Elliptic Curve cryptography algorithm, which is gaining popularity in applications that use devices with limited processing capabilities, such as smartphones.
3. Implementing a similar neural network for data encryption using the RSA, which requires random seeds instead of one in DSA. This can improve the security of data communications as the secret keys are not stored in any storage unit, hence, only the user that the information is directed to can decrypt it.

REFERENCES

[1] C. Liu and K. P. Arnett, "Exploring the factors associated with Web site success in the context of electronic commerce," Information & management, vol. 38, pp. 23-33, 2000.

[2] S. Zeadally, A. K. Das, and N. Sklavos, "Cryptographic technologies and protocol standards for Internet of Things," Internet of Things, p. 100075, 2019.

[3] R. K. Kanaan, G. Abumatar, A. M. A. Hussein, and M. Al-Lozi, "Management Information System using Blockchain Technology in an E-commerce Enterprise: A Systematic Review," Journal of Business & Management (COES&RJ-JBM), vol. 7, pp. 216-233, 2019.

[4] K. Balasubramanian and M. Rajakani, "Electronic Payment Systems and Their Security," in Digital Currency: Breakthroughs in Research and Practice, ed: IGI Global, 2019, pp. 270-285.

[5] T. Jisha and T. Monoth, "Authenticity and Integrity Enhanced Active Digital Image Forensics Based on Visual Cryptography," in Smart Intelligent Computing and Applications, ed: Springer, 2019, pp. 189-196.

[6] S. Kota, V. N. R. Padmanabhuni, K. Budda, and K. Sruthi, "Authentication and Encryption Using Modified Elliptic Curve Cryptography with Particle Swarm Optimization and Cuckoo Search Algorithm," Journal of The Institution of Engineers (India): Series B, vol. 99, pp. 343-351, 2018.

[7] A. H. A. Othman, S. M. Alhabshi, and R. Haron, "The effect of symmetric and asymmetric information on volatility structure of crypto-currency markets: A case study of bitcoin currency," Journal of Financial Economic Policy, 2019.

REFERENCES

[8] D. Schinianakis and T. Stouraitis, "RNS-Based Public-Key Cryptography (RSA and ECC)," in Embedded Systems Design with Special Arithmetic and Number Systems, ed: Springer, 2017, pp. 311-344.

[9] S. Sciancalepore, G. Piro, G. Boggia, and G. Bianchi, "Public key authentication and key agreement in IoT devices with minimal airtime consumption," IEEE Embedded Systems Letters, vol. 9, pp. 1-4, 2016.

[10] M. Thangapandiyan, P. R. Anand, and K. S. Sankaran, "Quantum Key Distribution and Cryptography Mechanisms for Cloud Data Security," in 2018 International Conference on Communication and Signal Processing (ICCSP), 2018, pp. 1031-1035.

[11] P. Gallagher, "Digital signature standard (dss)," Federal Information Processing Standards Publications, volume FIPS, pp. 186-3, 2013.

[12] C. F. Kerry and P. D. Gallagher, "Digital signature standard (DSS)," FIPS PUB, pp. 186-4, 2013.

[13] S. P. Banerjee and D. L. Woodard, "Biometric authentication and identification using keystroke dynamics: A survey," Journal of Pattern Recognition Research, vol. 7, pp. 116-139, 2012.

[14] G. L. Masala, P. Ruiu, and E. Grosso, "Biometric authentication and data security in cloud computing," in Computer and Network Security Essentials, ed: Springer, 2018, pp. 337-353.

[15] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 815-823.

REFERENCES

[16] M. Kumar, T. Garfinkel, D. Boneh, and T. Winograd, "Reducing shoulder-surfing by using gaze-based password entry," in Proceedings of the 3rd symposium on Usable privacy and security, 2007, pp. 13-19.

[17] Labati, R. D., et al, "Deep-ECG: convolutional neural networks for ECG biometric recognition." Pattern Recognition Letters 126: 78-85. (2019).

[18] S. Zhang, X. Ou, and D. Caragea, "Predicting cyber risks through national vulnerability database," information Security Journal: A Global Perspective, vol. 24, pp. 194-206, 2015.

[19] Y. H. Ali and I. S. A. Aljabar, “Real Time Face Recognition in Video Using Linear Discriminate Analysis and Local Binary Patterns,” Eng. Technol. J., vol. 33, no. 4 Part (B) Scientific, pp. 690–701, 2015.

[20] M. S. I. Sameem, T. Qasim, and K. Bakhat, “Real time recognition of human faces,” ICOSST 2016 - 2016 International Conference on Open Source Systems and Technologies, Proceedings. pp. 62–65, 2017.

[21] L. Siwik and L. Mozgowoj, "Server-side encrypting and digital signature platform with biometric authorization," International Journal of Computer Network and Information Security, vol. 7, p. 1, 2015.

[22] S. Haji and A. Varol, “Real time face recognition system (RTFRS),” 4th International Symposium on Digital Forensic and Security (ISDFS), 2016, pp. 107–111, 2016.

[23] Rahmawati, E., et al ,Digital signature on file using biometric fingerprint with fingerprint sensor on smartphone. International Electronics Symposium on Engineering Technology and Applications (IES-ETA), IEEE , (2017).

REFERENCES

[24] P. Lozhnikov and A. Sulavko, "Generation of a biometrically activated digital signature based on hybrid neural network algorithms," in Journal of Physics: Conference Series, 2018, p. 012047.

[25] Chen, F.-L., et al ."Public-key quantum digital signature scheme with one-time pad private-key." Quantum Information Processing 17(1): 10, (2018).

[26] Kotov, M., et al, "An attack on the Walnut digital signature algorithm." Designs, Codes and Cryptography 87(10): 2231-2250 , (2019).

[27] Chen, F.-L., et al, "Public-key quantum digital signature scheme with one-time pad private-key." Quantum Information Processing 17(1): 10, (2018).

[28] S. A. Jaju and S. S. Chowhan, “A Modified RSA algorithm to enhance security for digital signature,” in *2015 International Conference and Workshop on Computing and Communication (IEMCON)*, 2015, pp. 1–5.

[29] R. Kaur and A. Kaur, “Digital signature,” in *2012 International Conference on Computing Sciences*, 2012, pp. 295–301.

[30] M. B. Yassein, S. Aljawarneh, E. Qawasmeh, W. Mardini, and Y. hamayseh, “Comprehensive study of symmetric key and asymmetric key encryption algorithms,” in *2017 international conference on engineering and technology (ICET)*, 2017, pp. 1–7.

[31] F. J. Aufa and A. Affandi, “Security System Analysis in Combination Method: RSA Encryption and Digital Signature Algorithm,” in *2018 4th International Conference on Science and Technology (ICST)*, 2018, pp. 1–5.

REFERENCES

[32] R. Roshdy, M. Fouad, and M. Aboul-Dahab, “Design And Implementation A New Security Hash Algorithm Based On Md5 And Sha-256,” *Int. J. Eng. Sci. Emerg. Technol.*, vol. 6, no. 1, pp. 29–36, 2013.

[33] R. Martino and A. Cilardo, “A Flexible Framework for Exploring, Evaluating, and Comparing SHA-2 Designs,” *IEEE Access*, 2019.

[34] N. Saxena and S. Grijalva, “Efficient signature scheme for delivering authentic control commands in the smart grid,” *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 4323–4334, 2017.

[35] S. R. A. H. MIR, Z. A. JHAT, "Biometrics Verification: a Literature Survey," Journal of Computing and ICT Research, vol. 5, p. 14, 2011.

[36] G. J. Mohammed, "Efficient Iris Segmentation Method For Recognition Based on New Projection Function " Doctor of Engineering School of Computer Science and Technology, Harbin Institute of Technology, 2011.

[37] Kalyani, C. H. (2017). Various biometric authentication techniques: a review. J Biom Biostat, 8(5), 1–5.

[38] A. A. Jarjes, "Improving Snake Models For Accurate And Efficient Iris Segmentation," Doctor of Engineering, School of Computer Science and Technology, Harbin Institute of Technology, 2011.

[39] R. R. Debnath Bhattacharyya, Farkhod Alisherov A., and Minkyu Choi, "Biometric Authentication: A Review," International Journal of u- and eService, Science and Technology, vol. 2, p. 16, 2009.

REFERENCES

[40] A. Kumar and K. V. Prathyusha, "Personal authentication using hand vein triangulation and knuckle shape," IEEE Transactions on Image Processing, vol. 18, pp. 2127-2136, 2009.

[41] S. Wadhankar, P. Singh, and S. Sahoo, “Real Face Detection and Recognition: The Live Experiment,” Int. J. Comput. Appl., vol. 180, pp. 20–27, Mar. 2018.

[42] K. Dharavath, F. A. Talukdar, and R. H. Laskar, “Improving face recognition rate with image preprocessing,” Indian J. Sci. Technol., vol. 7, no. 8, pp. 1170–1175, 2014.

[43] E. Jose, M. Greeshma, M. H. TP, and M. H. Supriya, “Face Recognition based Surveillance System Using FaceNet and MTCNN on Jetson TX2,” in 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS), 2019, pp. 608–613.

[44] M. et al Kalamani, “Image Scaling Processor using Bilinear Algorithm M,” Int. Conf. Innov. Information, Embed. Commun. Syst., vol. 0975, no. 8887, 2014.

[45] M. Wang, Z. Wang, S. Zhang, J. Luan, and Z. Jiao, Face Expression Recognition Based on Deep Convolution Network. 2018.

[46] Y. Zhu, R. Dai, G. Liu, Z. Wang, and S. Lu, Power Market Price Forecasting via Deep Learning. 2018.

[47] N. Sahla, “A Deep Learning Prediction Model for Object Classification,” 2018.

[48] Hendrycks, D., et al, Using self-supervised learning can improve model

robustness and uncertainty. Advances in Neural Information Processing System, (2019).

[49] G. Kambourakis, D. Damopoulos, D. Papamartzivanos, and E. Pavlidakis, “Introducing touchstroke: keystroke-based authentication system for smartphones,” Security and Communication Networks, vol. 9, no. 6, pp. 542–554, 2016.

[50] J. Wayman, A. Jain, D. Maltoni, and D. Maio, “An introduction to biometric authentication systems,” in Biometric Systems, Springer, pp. 1–20, 2005.

[51] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” Information Processing & Management, vol. 45, no. 4, pp. 427–437, 2009.

[52] M. O’Reilly et al., Binary Classification of Running Fatigue using a Single Inertial Measurement Unit. 2017

[53] Tang, Y., et al, "Principal curvature measures estimation and application to 3D face recognition." Journal of Mathematical Imaging and Vision 59(2): 211-233. (2017).

[54] O. Z. Akif, “Secure authentication procedures based on timed passwords, honeypots, honeywords and multi-factor techniques.” Brunel University London, 2017.

[55] P. J. Phillips, A. Martin, C. Wilson, and M. Przybocki, "An introduction to evaluating biometric systems," Computer, pp. 56-63, 2000.

الخلاصة

يحظى التوقيع الرقمي باهتمام كبير في السنوات الأخيرة ، وفقا لسرعة تزايد استخدام المعلومات الرقمية الوصول إلى العديد من الخدمات. تعتمد مسؤولية نظام التوقيع الرقمي بشكل أساسي على أمان المفاتيح السرية للمستخدمين ، حيث يتم استخدام هذه المفاتيح لإثبات صحة البيانات المستلمة وسلامتها. علاوة على ذلك ، طالما أن المفتاح السري للمرسل غير معروف لأي فرد آخر ، مثل المهاجمين ، فلا يمكن للمرسل أن يتنصل من أن المستند مصدره. وبالتالي ، تم اقتراح طريقة جديدة في هذه الدراسة لاستخراج المفاتيح السرية للمستخدمين مباشرة من ميزات الوجه الخاصة بهم ، دون تخزينها في قاعدة البيانات. وبالتالي ، يتطلب توقيع مستند وجودًا فعليًا من المرسل ، ولا يسمح اختراق قاعدة البيانات للمهاجم بإنتاج توقيعات خاطئة. وبالتالي ، فإن مسؤولية النظام المقترح أعلى بكثير من أي نظام آخر موجود في الاعمال السابقه .

يتبع النظام المقترح معيار التوقيع الرقمي(DSS) باستخدام دالة التجزئة SHA 256 إنتاج تجزئة فريدة لكل وثيقة رقمية يتم توقيعها و خوارزميه التوقيع الرقمي (DSA) لتشفير والتحقق من صحة تجزئة المستندات المرسلة والمستلمة. يتطلب المفتاح السري لأحد المستخدمين عددًا أوليًا من بعض الخصائص. تستخدم الطريقة المقترحة شبكة عصبية لإنتاج بذرة عشوائية تُستخدم لتوليد العدد الأولي المطلوب. لتدريب الشبكة العصبية لإنتاج بذور عشوائية قوية وفريدة من نوعها ، يتم استخدام نهج تدريب شبه خاضع للإشراف. يعتمد هذا النهج على القيم التي تنبأت بها الشبكة العصبية لكل فرد لتحديد القيمة المثلى لذلك الفرد. وبالتالي ، فإن القيمة المخصصة لكل مستخدم تعكس الميزات التي تعثر عليها الشبكة العصبية في ميزات الوجه ، بدلاً من فرض القيم التي قد لا تكون مناسبة للميزات المكتشفة.

نظرًا لأن القيم التي تنتجها الشبكة العصبية لتوليد البذور العشوائي ليست مضمونة لتكون فريدة ومتينة لكل مستخدم ، يتم التحقق من صحة المفاتيح السرية التي تم إنشاؤها باستخدام المفاتيح العامة للمستخدم المخزنة في قاعدة البيانات. بالإضافة إلى ذلك ، لتحسين أمان النظام المقترح وتحصينه ضد بعض الهجمات ، مثل الخداع ، يتم تدريب شبكة عصبية لاكتشاف حيوية مقاطع الفيديو التي تم جمعها للمستخدمين في الوقت الفعلي. علاوة على ذلك ، لتأمين الطريقة المقترحة ضد هجمات الكتف

والتصيد ، فإن المستخدمين مطالبون بالمصادقة باستخدام مقاطع الفيديو الحية التي تم التقاطها لوجوههم بعد المصادقة باستخدام أسماء المستخدمين وكلمات المرور الخاصة بهم .

يتم تجميع مجموعة من 30 صورة وجهية في الوقت الفعلي من المستخدم وتستخدم في كل مرحلة من المراحل على الطريقة المقترحة. تظهر النتائج التجريبية التي أجريت باستخدام مقاطع فيديو في الوقت الفعلي تم جمعها من 33 متطوعًا ، أن تسجيل مستخدمي النظام المقترح يتطلب في المتوسط 1.082 مجموعة دفع لكل مستخدم و 4.285 لتوقيع المستندات. أظهرت القيم التي تنتجها الشبكة العصبية لتوليد البذور العشوائية 19.72 ٪ متانة و 100 ٪ التفرد ، وذلك باستخدام أشرطة الفيديو نفسها التي تم جمعها من المتطوعين. علاوة على ذلك ، أظهرت الشبكة العصبية لاكتشاف الكشف عن درجة 99.96 ٪ F1 ، في حين حققت طريقة مصادقة المستخدم 93.92 ٪.

وزارة التعليم العالي والبحث العلمي

جامعة ديالى - كلية العلوم

قسم علوم الحاسوب

نظام التوقيع الرقمي القائم على التعرف على الوجه ثلاثي الأبعاد في الوقت الحقيقي

رسالة

مقدّمة الى قسم علوم الحاسوب – كلية العلوم – جامعة ديالى كجزء من متطلبات نيل درجة الماجستير في إختصاص علوم الحاسوب

من قبِل

اسراء صفاء احمد

بإشراف

أ.م.د طه محمد حسن

أ.م.د فراس عبد الحميد عبد الطيف

1440 **2019**