ENHANCED FACE LIVENESS DETECTION BASED ON FEATURES FROM NONLINEAR DIFFUSION USING SPECIALIZED DEEP CONVOLUTION NETWORK AND ITS APPLICATION IN OAUTH

Aziz Alotaibi

Under the Supervision of Dr. Ausif Mahmood

DISSERTATION
SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE

AND ENGINEERING

THE SCHOOL OF ENGINEERING

UNIVERSITY OF BRIDGEPORT

CONNECTICUT

December, 2016

ENHANCED FACE LIVENESS DETECTION BASED ON FEATURES FROM NONLINEAR DIFFUSION USING SPECIALIZED DEEP CONVOLUTION NETWORK AND ITS APPLICATION IN OAUTH

Aziz Alotaibi

Under the Supervision of Dr. Ausif Mahmood

Approvals

Committee Members

Name	Signature	Date
Dr. Ausif Mahmood	Ausif Mahmood	12-21-2016
Dr. Navarun Gupta		12/21/16
Dr. Prabir Patra		12/21/16
Dr. Maid Faezipour		12,21,2016
Dr. Saeid Moslehpour		12,14,16

Ph.D. Program Coordinator

Dr. Khaled M. Elleithy		12/21/2016

Chairman, Computer Science and Engineering Department

Dr. Ausif Mahmood	Ausif Mahmood	12-21-2016

Dean, School of Engineering

Dr. Tarek M. Sobh		12/21/2016

ENHANCED FACE LIVENESS DETECTION BASED ON FEATURES FROM NONLINEAR DIFFUSION USING SPECIALIZED DEEP CONVOLUTION NETWORK AND ITS APPLICATION IN OAUTH

ENHANCED FACE LIVENESS DETECTION BASED ON FEATURES FROM NONLINEAR DIFFUSION USING SPECIALIZED DEEP CONVOLUTION NETWORK AND ITS APPLICATION IN OAUTH

ABSTRACT:

The major contribution of this research is the development of enhanced algorithms that will prevent face spoofing attacks by utilizing a single image captured from a 2-D printed image or a recorded video. We first apply a nonlinear diffusion based on an additive operator splitting (AOS) scheme with a large time step to acquire a diffused image. The AOS-based scheme enables fast diffusion that successfully reveals the depth information and surface texture in the input image. Then a specialized deep convolution neural network is developed that can extract the discriminative and high-level features of the input diffused image to differentiate between a fake face and a real face. Our proposed method yields higher accuracy as compared to the previously implemented state-of-the-art methods. As an application of the face liveness detection, we develop face biometric authentication in an Open Authorization (OAuth) framework for controlling secure access to web resources. We implement a complete face verification system that consists of face liveness detection followed by face authentication that uses Local Binary Pattern as features for face recognition. The entire face authentication process consists of four services: an image registration service, a face liveness detection service, a verification service, and an access token service for use in OAuth.

ACKNOWLEDGEMENTS

My thanks are wholly devoted to God who has helped me all the way to complete this work successfully. I owe a debt of gratitude to my family for understanding and encouragement.

I would like to express my deepest appreciation to my advisor Dr. Ausif Mahmood for his guidance and for his tremendous help and support.

I would like to thank my dissertation committee of Dr. Navarun Gupta, Dr. Prabir Patra, Dr. Maid Faezipour, Saeid Moslehpour and Dr. Khaled M. Elleithy for their time and valuable comments that helped me to improve the quality of this dissertation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

1.1.Research Problem and Scope

Face recognition authentication is vulnerable to several attacks especially face spoofing attacks. A spoofing attack manipulates the system by presenting a forgery to the acquisition sensor with the goal of penetrating the biometric authentication system. More specifically, a face-spoofing attack can be accomplished by presenting a 2D image, digital video or 3D mask to the camera, mimicking the user and thus gaining access as a valid user. This vulnerability has induced researchers to propose several countermeasures to prevent face-spoofing attacks. These countermeasures are intended to detect the “liveness” of the face before performing the face recognition operation. Such anti-spoofing approaches can be further classified into two groups: static techniques and dynamic techniques. The static techniques are based on the analysis of a 2D single static photograph. In contrast, dynamic techniques are based on analyzing the temporal features of a sequence of input frames. Dynamic techniques are slow and difficult to implement. Furthermore, some of the dynamic techniques require users to follow instructions to validate their presence, but not all users may cooperate in this respect. This makes the dynamic methods unfavorable.

OAuth protocol implements traditional credentials to authenticate the resource owner which makes the security of users' information at risk. Therefore, creating reliable, scalable, and maintainable systems has become an essential core of function of security. Several security methods have been developed to authenticate users' identities, including knowledge-based methods and ownership-based methods. These methods are commonly implemented in online user authentication to control access to users' data and verify their identities. However, knowledge-based methods (e.g., username/password, secret questions) are vulnerable to attacks, such as the man-in-the-middle attack, the replay attack, and stolen-verifier attacks. In contrast, ownership-based methods are based on something the user owns, such as a smart card or a token that can be reused, stolen, or manipulated. In both knowledge and ownership methods, the authentication system verifies what the user knows or possesses rather than truly verifying the identity of the requester. As an alternative, biometric authentication verifies the identity of requesters by using their physiological and/or behavioral characteristics.

1.2.Motivation Behind the Research

The adoption of cloud computing and web services-based software architectures has grown rapidly, and these technologies have played a vital role in the information technology field [1] [2]. Secure access to the web API has become more important and gets more complex with the increasing adoption of cloud and web services. Controlling the user level to access the right resources has brought several security issues to the web API field. One of the most popular API access control models that has been proposed is a standard known as open authorization (OAuth) [3]. OAuth uses traditional credentials as an authentication method for different applications and services. In OAuth, the authorization server is responsible for managing and securing users' information as well as issuing the access token. Since the majority of OAuth providers are implementing traditional credentials to authenticate the resource owner, the entire system is exposed to dictionary and brute force attacks. If either of these attacks should occur, the attackers have access to the user's system, and thus can grant their third-applications access to all register services. To overcome this issue, biometric authentication cabaple of providing security. A system that employs biometric authentication ideally exhibits five qualities: robustness, distinctiveness, availability, accessibility and acceptability [4]. In online user authentication, accessibility and acceptability are the most significant qualities that can be found in face and voice characteristics. However, face recognition is commonly favored over other biometric traits due to its accessibility and non-intrusive form of interaction. Face recognition has been actively explored and researched in the field of security. However, any photograph of a valid user (easily obtained by capturing a close-up photograph without the user's consent or obtained via the Internet) can be used to spoof face recognition systems. Therefore, developing a face recognition authentication service as an authentication mechanism is essential to secure web services and OAuth.

1.3. Potential Contributions of the Proposed Research

We developed an efficient and non-intrusive method to counter face spoofing attacks that uses a static print photograph, or recorded video, of a valid user. We first apply a nonlinear diffusion based on an additive operator splitting (AOS) scheme to acquire a diffused image with a large time step value. To extract the information features from the diffused image, previous approaches used handcrafted features, such as the LBP algorithm. In contrast, this dissertation developed a specialized deep convolution neural network (CNN) that can extract the discriminative and high-level features to differentiate between a fake face and a real face. Our developed CNN architecture was able to extract the most significant features from the diffused image. We achieved the highest reported accuracy of 99% on the widely-used NUAA dataset. In addition, we tested our method on the Replay Attack dataset which consists of 1200 short videos of both real-access and spoofing attacks. An extensive experimental analysis was conducted, and demonstrated better results when compared to previous static algorithms results.

In addition, we introduced a Secure Login Service to enhance OAuth security using face recognition in order to authenticate the identity of the user. The Secure Login Service process consists of four services: an image registration service, a face liveness detection service, a verification service, and an access token service. We built these services in the authorization server to verify the user and issue the access token. The entire web services are built based on the RESTful architecture style. Two preprocesses occur prior to using Local Binary Pattern (LBP) texture method. The first preprocess localizes five landmark points: the outer right eye, center right eye, outer right eye, center left eye, and the center of nose. While the second preprocess performs the alignments. The LBP texture method is used to extract the image features, and then apply the correlation function to classify the captured image and decide whether to verify or deny the user. If the user is verified and authenticated, the access token is issued with a specific scope of authorization level. This

Secure Login Service enhances OAuth security by providing biometric authentication. This can be used in addition to the regular authentication.

CHAPTER 2: LITERATURE SURVEY

2.1 Face Liveness Detection

2.1.1 Introduction to Anti-Spoofing Methods

Recently, the performance of the face recognition system has been enhanced significantly because of improvements found within hardware and software techniques in the computer vision field [5]. However, face recognition is still vulnerable to several attacks such as spoofing attacks. These techniques are getting more complex and hard to identify, especially with the advancement in printer technology such as high-definition laser printers. Therefore, researchers have proposed and analyzed several approaches to protect face recognition systems against these vulnerabilities. Based on the proposed techniques, face anti-spoofing methods are grouped into two main categories: hardware-based technique and software-based technique. First, hardware-based technique requires an extra device to detect a particular biometric trait such as finger sweat, blood pressure, facial thermogram, or eye reflection [6]. This sensor device, incorporated into the biometrics authentication system, requires the user's cooperation to detect the signal of the living body. Some auxiliary devices, such as infrared equipment, achieve higher accuracy when compared to simpler devices. However, auxiliary devices are expensive and difficult to implement [7]. Second, the software-based technique extracts the feature of the biometric traits through a standard sensor to distinguish between real and fake traits. The feature extraction occurs after the biometric traits, such as the texture features in the facial image, are acquired by the sensor [8]. The software-based techniques treat both the acquired 3D and 2D traits as 2D to extract the information feature. Therefore, the depth information is utilized to differentiate between 3D live face and flat 2D fake face images [9]. This dissertation covers only the software-based techniques that can be categorized further into static-based techniques and dynamic-based techniques as described in the following sections.

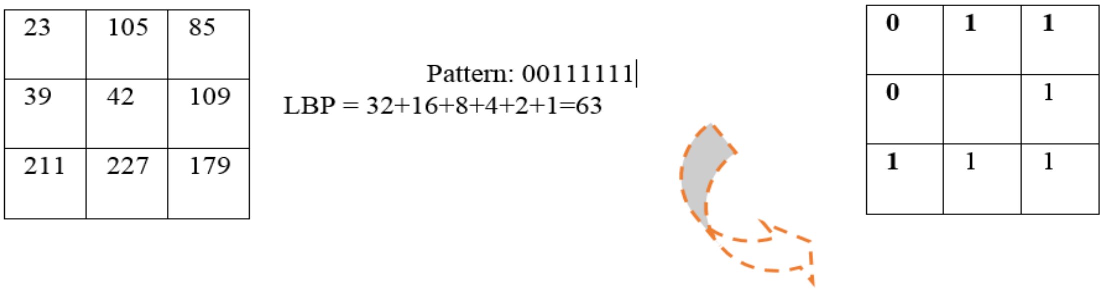
2.1.2 Software based techniques

Static-based and dynamic-based techniques are less expensive and easy to implement compared to hardware-based techniques. First, static techniques are based on the analysis of a 2D static image. It is a non-intrusive interaction which is convenient for many users. On other hand, dynamic techniques exploit the temporal and spatial features using a sequence of input frames. Some of the dynamic methods are intrusive interactions which force the user to follow specific instructions.

2.1.2.1 Static technique

A variety of proposed methods are presented to address spoofing attack problems that utilize a single static image. The static-based techniques are divided into two categories: texture analysis methods and fourier spectrum methods:

Texture analysis methods: these methods extract the texture properties of the facial image based on the feature descriptor. Maatta et al. [10] analyzed the texture of a 2D facial image using multi-scale local binary pattern (LBP) to detect face liveness. The authors applied multi- LBP operators on the 2D face image to generate a concatenated feature histogram. The histogram is then fed into the Support Vector Machine (SVM) classifier in order to determine whether the facial image is real or fake. The Local Binary Pattern (LBP), introduced by Ojala et al. [11], is a nonparametric method that extracts the texture properties of the 2D facial image with features based on the local neighborhood [12] as shown in Figure 2.1. The basic LBP pattern operator for each pixel in the facial image is calculated by using the circular neighborhood as shown in Figure 2.1.



23	105	85
39	42	109
211	227	179

Pattern: 00111111

LBP = 32+16+8+4+2+1=63

0	**1**	**1**
0		1
1	1	1

Figure 2.1 The basic LBP Operator.

The intensity of the centered pixel is compared with the intensity value of the pixels located within its LBP 3*3 neighborhood.

$$LBP_{Point,Radius}(x_c + y_c) = \sum_{P=0}^{P-1} S(i_p - i_c)2^p \quad (2.1)$$

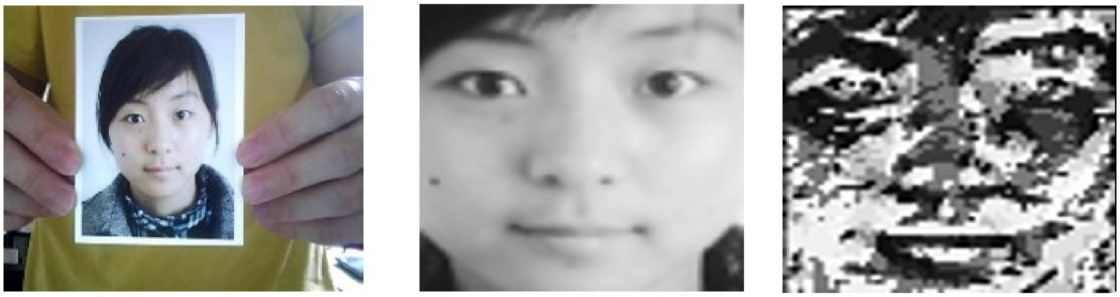
Where

xc,yc represent the center pixel

p represents the surrounding pixel

$$s(z) = \begin{cases} 1, & if\ z \geq 0 \\ 0, & if\ z < 0 \end{cases}$$

Then, the center pixel will be updated with the new pixel value of 63. The LBP uses a uniform pattern to describe the texture image. If the generated binary number contains at most two bitwise 0 -1 or vice versa, then the LBP is called uniform. For instance, (01111110), (1100 0000), and (0001 1000) are uniform, whereas (0101 000), (0001 0010), and (0100 0100) are non-uniform. There are 58 uniform LBP Patterns and 198 non-uniform LBP patterns. The authors applied three multi-scale LBP operators on the normalized face images: LBP 8,1 u2 , LBP 8,2 u2, and LBP 16,2 u2.



a) input image b) normalized face c) LBP image

Figure 2.2 Applying LBP operator on normalized face image.

LBP 8,1 u2 was applied on a nine-block region of the normalized face, and therefore, generated uniform patterns with a 59 –bin histogram from each region. The entire image equaled to a single 531-bin histogram. The LBP 8,2 u2, and LBP 16,2 u2 operators generates 59-bin and 243-bin histogram, respectively. The length of the concatenated feature histogram is 833. The concatenated histogram is passed through a nonlinear SVM classifier to determine whether the input face image is present or not. However, the basic

LBP operator is not the only operator applied to extract the information features, other LBP variations might be used as well such as transitional (tLBP), direction-coded (dLBP) and modified (mLBP). In [13], Chingovska et al. introduced Replay-Attack Database and studied the effectiveness of the local Binary Pattern on three types of attacks: printed photographs, photos, and video displays.

Figure 2.3 A frame of short videos from.

The authors applied different LBP operators and studied the performance evaluation of the anti-spoofing algorithm. The study included tLBP, dLBP and mLBP. The tLBP operator is composed by comparing the two consecutive pixels' value with their neighbors in a clockwise direction for all pixels apart from the central pixel value.

$$LBP_{p,R}(x_c + y_c) = S(i_0 - i_{p-1}) + \sum_{P=0}^{P-1} S(i_p - i_{p-1})2^p \tag{2.2}$$

A direction-coded LBP operator is composed by comparing the intensity variation along the four base directions into two bits through the central pixel.

Let's assume the original LBPP, R has P =2P' neighbors.

$$dLBP_{p,R} = \sum_{P=0}^{P\prime-1}(S(i_{p\prime} - i_c)(i_{p\prime} + p' - i_c)2^{2p'} + S(|i_{p\prime} - i_c| - |i_{p\prime} + p' - i_c |2^{2p'+1}) \tag{2.3}$$

The dLBP compares the intensity of each pixel value of neighbors with the average of the intensity value in a 3 *3 neighborhood.

$$LBP_{Point,Radius}(x_c + y_c) = \sum_{P=0}^{P-1} S(i_p - i_c)2^p \tag{2.4}$$

Where

- x_c,y_c represent the center pixel
- p represents the surrounding pixel
- $s(z) = \begin{cases} 1, & if\ z \geq Ave \\ 0, & if\ z < Ave \end{cases}$

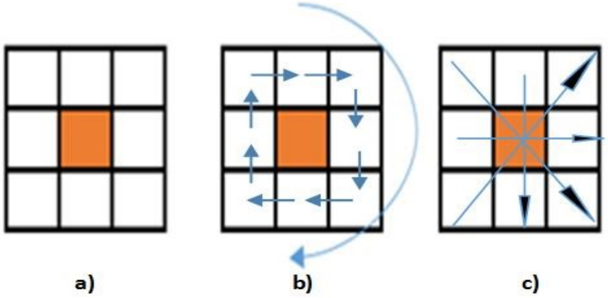


Figure 2.4 a) Modified b) Transition c) Direction LBP.

After applying the LBP Operators on the facial images, histograms are obtained as feature vectors. Then the applied classifier extracts the feature and determines whether the facial image is real or fake. Both linear and non-linear classifiers were examined, such as Linear Discriminant Analysis (LDA) and Support Vector Machine (SVM). The authors conducted an experiment to compare X2 statistics methods to other complex classifiers.

Table 2. 1 HTER (%) of the classification on different database

	REPLAY-ATTACK	NUAA	CASIA-FASD
LBP $_{3*3}^{u2}$ + X^2	34.01	-	-
LBP $_{3*3}^{u2}$ **+ LDA**	17.17	18.32	21.01
LBP $_{3*3}^{u2}$ **+ SVM**	15.16	19.03	18.17
LBP + SVM	13.87	13.17	18.21

From Table 2.1, we observed that the LBP extracts adequate features from the single static image, which assists in the classification of fake or real faces. The performance of the multi-scale LBP is calculated using the Half Total Error Rate (HTER). HTER is defined as half of the sum of the False Rejection Rate (FRR) and False Acceptance Rate (FAR). HTER is used to measure the performance on both the development and test sets. Both LDA and SVM show high performance on the development sets and low performance on the test sets.

$$\mathbf{HTER} = \frac{\mathbf{FRR} + \mathbf{FAR}}{2} \qquad (2.5)$$

Where,
FRR = FR/ NR *False Rejection, and Number of Real.*
FAR = FA/ NI *False Acceptance, and Number of Imposter.*

Table 2. 2 HTER (%) of classification with (X2) for different LBP operators on Replay-Attack Database.

LBP $_{3*3}{}^{u2}$		***tLBP***		***dLBP***		***mLBP***	
Dev	Test	Dev	Test	Dev	Test	Dev	Test
31.24	34.01	29.37	35.35	36.71	40.26	32.29	33.68

In [14] Kim et al. the authors proposed a real-time and non-intrusive method based on the diffusion speed of a single image to detect face liveness. Their idea is based on the difference in the illumination characteristic of both live and fake faces. The additive operator splitting (AOS) schema is used to compute the image diffusion [15]:

$$u^{k+1} = \frac{1}{2} ((I - 2\,\pi\, A_x(u^k)^{-1} + ((I - 2\,\pi\, A_y(u^k)^{-1})u^k \tag{2.6}$$

Where A_x and A_y denote the diffusion matrices computed in column wise and row wise. The AOS schema treats every coordinate axis in the same manner, and it is unconditionally stable with large time step, e.g. $\pi = 40$.

To compute the diffusion speed at each pixel position (x, y):

$$s(x, y) = |\log(u^0(x, y) + 1) - \log(u^L(x, y) + 1)| \tag{2.7}$$

The features are extracted using the Local pattern of the diffusion speed, the so-called Local Speed Pattern (LSP):

$$\mathrm{LSP}\ (\mathrm{x}, \mathrm{y}) = \sum_{1 \leq i \leq n} 2^{i-1}\, LSP^i\ (x, y) \tag{2.8}$$

$$LSP(x\,, y) \ = \begin{cases} 1, & if\ s(x, y) > (x_i\,, y_i) \\ 0, & otherwise, \end{cases} \tag{2.9}$$

Where n represents the number of sampling pixels. (x, y) is the center pixel, and (x_i, y_i) denotes the position neighborhood. The extracted feature is fed into the SVM classifier to determine whether the input face is real or fake.

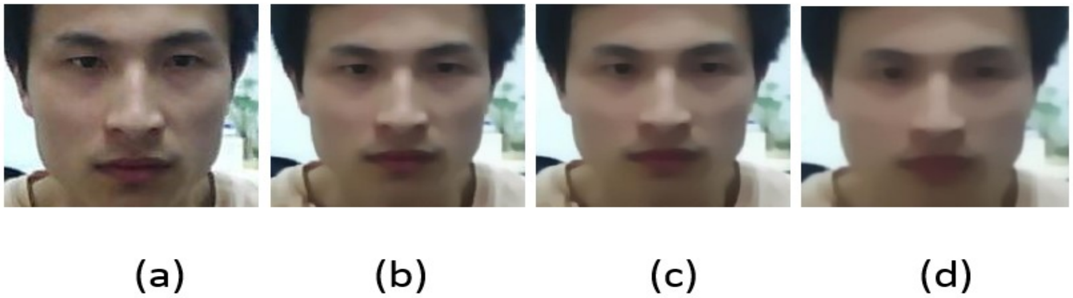


Figure 2.5 Example of diffusion image u^k with different iteration number and time step equals to 10. (a) original image. (b) k = 5. (c) k = 10. (d) k = 20.

Yang et al. [16] proposed a component-based face recognition coding approach for face liveness detection. First, the holistic face (H-Face) is divided into six components: the counter, facial, left eye, right eye, mouth, and nose regions. Subsequently, counter, and facial regions are further divided into 2 * 2 grids, respectively. Moreover, the dense low-level features such as LBP, LQP, HOG, etc. are extracted for all twelve components. Furthermore, component-based coding is performed to derive high-level face representation of each one of the twelve components from low-level features. Finally, the concatenating histograms from all twelve components are fed to a SVM classifier for identification. They achieved good accuracy compared with most of the proposed methods as shown in table 2.3.

Table 2. 3 Performance on NUAA, PRINT-ATTACK, and CASIA

Database	Scenario	Accuracy with Metric (5)
NUAA	--	**0.977**
PRINT-ATTACK	Fixed(F) sub-database	**0.995**
	Hand (H)sub-database	**0.991**
	(F) and(H)sub-databases	**0.988**
CASIA	Low Quality	**0.987**
	Low Quality	**0.931**
	Warped Photo	**0.930**
	Video Photo	**0.997**
	Overall test	**0.898**

To sum up, the texture analysis methods are used to extract the discriminative features for texture based classifications. However, they are less sensitive to noise in uniform regions, and their performance is degraded under changing lighting directions and shadowing [17].

Methods based on Fourier spectra: Fourier spectra is used to capture the frequency distribution of the input images to detect spoofing attacks. The structure texture of fake images are 2-D and real images are 3-D. The reflection of the light on 2D and 3D objects result in different frequency distribution. Therefore, the intensity contrast of fake images contains a smaller frequency component. In [18]-[19], the authors analyzed the input images using 2D Fourier spectra to extract the feature information in order to detect whether the input image is real or fake. Unlike [4]-[46], which used very high frequency band which is too noisy, the authors applied a Difference of Gaussian (DoG) filter in which two Gaussian filters with different standard deviation extract the difference of the image variability.

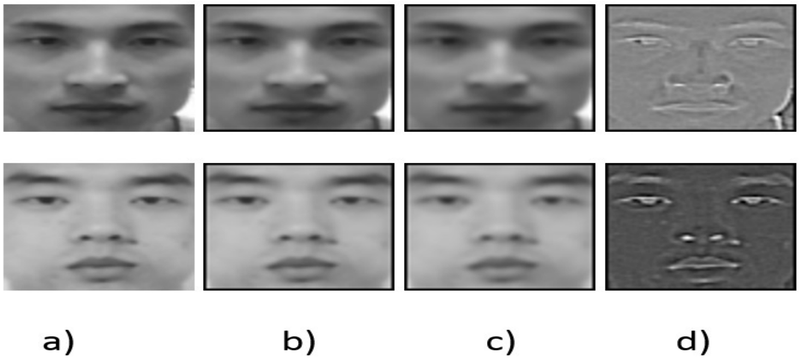


Figure 2.6 (a) input image, (b) Gσ1 =0.5 ,(c) Gσ2 =0.5, (d) Difference of Gaussian.

As depicted in Figure 2.6, DoG is applied to remove lighting variation in the input image and preserve as many features as possible without causing noise. Gaussian function with standard deviation σ1 as given:

$$G_{\sigma_1}(x, y) = \frac{1}{\sqrt{2\pi\sigma_1^2}} exp\left(-\frac{x^2 + y^2}{2\sigma_1^2}\right) \tag{2.10}$$

Table 2. 4. Gaussian filter (3, 3) with σ1 =1.0 and 0.5 respectively.

0.075	0.124	0.075
0.124	0.204	0.124
0.075	0.124	0.075

0.011	0.084	0.011
0.084	0.62	0.084
0.011	0.084	0.011

Gaussian filter g (u, v) with two different standard deviations σ1 =0.5, σ2 =1.0 on the input image f(x, y) is defined as:

$$\mathrm{DoG}\ (\mathrm{x}, \mathrm{y}) = \ \big(\mathrm{G}\sigma 1(\mathrm{u}, \mathrm{v}) * \ \mathrm{f}\ (\mathrm{x}, \mathrm{y})\big) - \\ (\ \mathrm{G}\sigma 2\ (\mathrm{u}, \mathrm{v})\ \ \mathrm{f}\ (\mathrm{x}, \mathrm{y}) \qquad (2.11)$$

Peixoto et al. [19] used DoG with the Sparse Logistic Regression Model to detect the spoofing attack under extreme illumination. The Sparse Logistic Regression is given as:

$$Prob(y|x) = \frac{1}{1 + \exp(-\mathrm{y}(w^T x + b))} \qquad (2.12)$$

Where w is the weight vector, b is the intercept, and the average logistic loss is defined as:

$$Loss(w, b) = \frac{1}{m}\sum_{i=0}^{m} \log(1 + \exp(-y_i\ w^T\ x_i + b)) \qquad (2.13)$$

The authors used the contrast-limited adaptive histogram equalization [20] to deal with the illumination changes, which affect the input image. In addition, Tan et al [18] applied the DoG and the variation Retinex-based to extract the latent reflectance features. The authors modified the Sparse Logistic Regression to learn a "low rank" projection matrix.

Table 2. 5. Experiment result for NUAA database.

Approach	**Min**	**Mean**	**Max**	**STD**
Tan et al "low rank" [18]	85.2%	86.6%	87.5%	0.6%
Peixoto et al "bad illumination" [19]	92.0%	93.2%	94.5%	0.4%

Table 2.5. shows that the DoG with the Sparse Logistic Regression achieved 94.5% on the NUAA dataset. This result reflects that the Fourier spectra methods have the ability to capture enough feature of the input image in order to identify the spoof attack. Further, Zang, et al [21] used multiple difference of Gaussian (DoG) filters to extract the high frequency feature from the input face image. Four DoG filters are used to compute the inner and outer Gaussian variance. Let σ1 represent the inner variance, σ2 the outer variance:

σ1 =0.5, σ2 = 1; σ1 =1.0, σ2 = 1.5; σ1 =1.5, σ2 = 2; and σ1 =1 , σ2 = 2.

Then the concatenated filtered images are fed into the SVM classifier. Moreover, Li et al. [22] detected the live and fake face images based on their analysis of their 2D Fourier Spectra on the face and [4] on the hair . The authors calculate the high frequency component using the high frequency descriptor equation. The high-frequency descriptor of a live face should be greater than a predefined threshold Tft , and the value of Fourier transform is more than the predefined threshold Tf.

$$HFD = \frac{\iint_{\Omega=\{(u,v) \mid \sqrt{u^2+v^2} > \frac{2}{3} f_{\max} \; and \, |F(u,v)| > T_f\}} |F(u,v)| \, dudv}{\iint |F(u,v)| \, dudv - F(0,0)} \text{ x } 1000$$

(2.14)

Where F(u, v) represents the fourier transform of the input image, fmax denotes the highest radius frequency of F(u, v), Tf, and Tfd are a predefined threshold. The denominator denotes the total energy in the frequency domain which is the sum of Fourier coefficients relative to the direct coefficient.

2.1.2.2Dynamic technique

Dynamic methods rely on the detection of motion over the input frames sequence to extract dynamic features enabling the distinction between real and fake faces. Pereira et al. [23] proposed a novel countermeasure against face spoofing based on the Local Binary Pattern from three Orthogonal Plans (LBP-TOP) which combine both space and time information into a multi-resolution texture descriptor. Volume Local Binary Pattern (VLBP) [24], which is an extension to the Local Binary Pattern, was introduced to extract the features from the dynamic texture.

$$VLBP_{L,P,R} = \sum_{q=0}^{3P+1} f(i_c - i_q)\, 2^q \tag{2.15}$$

And f(x) is defined:

$$f(x) = \begin{cases} 0 & if\ x < 0 \\ 1 & if\ x \geq 0 \end{cases}$$

VLBP considers the frame sequence as parallel sequence planes, unlike LBP-TOP which considers the three orthogonal planes intersecting the pixel of the center for each pixel in a frame sequence. Orthogonal planes consist of an XY plane, XT plane, and YT plane, where T represents the time axis. Three different histograms are generated from the three orthogonal planes and then concatenated and fed to the classifier. In [25] , Bharadwaj et al presented a new framework for face video spoofing detection using motion magnification. The Eulerian motion magnification technique is applied to enhance the facial expressions exhibited by clients in a captured video. In the feature extraction stage, the authors used both multi-scale LBP (LBP 8,1 u2 , LBP 8,2 u2 , and LBPu2 16,2), and the Histogram of Oriented Optical Flows (HOOF). The optical flow is the pattern of the apparent motion estimation technique that computes the motion of each pixel by solving the optimization problem. The PCA is used to reduce the dimensionality of the HOOF vector. Finally, an LDA classifier is used to classify the concatenated HOOF and detect whether the video input is real or face access.

Further, Pan et al. [26] proposed an eye-blinking behavior method to detect spoofing face recognition based on an unidirectional conditional graphic framework. The eye-blinking behavior is represented as temporal image sequences after being captured. The unidirectional conditional model reduces the computational cost. It is easy to extract the feature from the intermediate observation, where the conditional model increases the complexity and makes the problem more complicated. The authors developed an eye closity method by computing discriminative information for eye states:

$$u_m\,(I) = \sum_{i=1}^{M}(log\ \frac{1}{\beta_i})\ h_i(I) - \frac{1}{2}\sum_{i=1}^{M} log\ \frac{1}{\beta_i} \tag{2.16}$$

Where,

$$\beta_i = \epsilon_i/(1-\epsilon_i)$$

And $u(I)$ is the eye closity, and $h_i(I) : R^{d(i)} \rightarrow \{0,1\}, i = 1, 2\,, .., M\ \}$ is a set of binary weak classifier. The input I has two states, open eye: (0) and closed eye: (1) . β

represents a closing eye state. The Adaboost algorithm is used to classify the positive value as a closed eye and a negative value as an open eye. A blinking activity sequence of eye closity is shown in Figure 2.7.

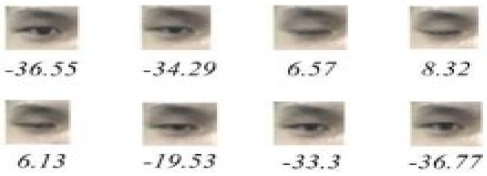


Figure 2. 1. Illustration of the closity for a blinking activity sequence.

In [27] Wen et al. proposed a face spoof detection algorithm based on Image Distortion Analysis (IDA). Four different types of IDA features (specular reflection, blurriness, color moments, and color diversity) have been extracted from the input frame. The IDA features are concatenated together to produce a 121-dimentional IDA feature vector. The feature vector is fed into an ensemble classifier; a multiple SVM classifier to distinguish between real and spoof faces. Their detection algorithm is extended to the multi-frame face detection in the playback video using a voting-based schema. IDA technique is computationally expensive and consumes time when using multi-frames to detect the spoofing attack.

In [28] , Singh et al. proposed a framework to detect the face liveness using eye and mouth movement. Challenge and response are randomly generated in order to detect and calculate the eye and mouth movements using Haar Classifier. The eye openness and closeness can be measured during the time interval while the mouth is measured using the teeth Hue Saturation Value (HSV). If the calculated response is equal to the number of the challenges, the proposed system will recognize the user as live.

Kim et al. [29] presented a novel method for face spoofing detection using camera focusing. Two sequential images were taken with two different focusing: on the nose (IN) and the ears (IE). SUM Modified Laplacian (SML) is used to measure the degree of focusing for both the nose (SN) and ears (SE). After calculating SMLs, the SN is subtracted from SE to maximize the SML gap between the nose and ears regions. If the sum of the difference of SMLs (DoS) shows similar pattern consistently, the user is live. Otherwise it is fake. The difference in the patterns can be used as features to detect the face liveness.

In [30], Kim et al. segmented the video input into the foreground and background regions to detect the motion and similarity in order to prevent image and video spoofing attacks. The authors used a structural similarity index measure (SSIM) to measure the similarity between the initial background region and the current background region. The background motion index (BMI) is proposed to show the amount of motion in the background compared with foreground region. The motion and similarity in the background region should contain significant information to indicate liveness detection.

In [31], Tirunagari et al. used a recently-developed algorithm called Dynamic Mode Decomposition (DMD) to prevent replay attacks. The DMD algorithm is a mathematical method developed to analyze and extract relevant modes from empirical data generated by non-linear complex fluid flows. The DMD algorithm can represent the temporal information of the entire input video as a single image with the same dimensions as those images contained in the recorded video. The authors modified the original MDM that uses QR-decomposition and used LU decomposition to make it more practical. The DMD is used to capture the dynamic visual in the input video. The feature information is extracted from the visual dynamic using the LBP and fed to the SVM classifier.

Yan et al. [32] proposed a novel liveness detection method based on three clues in both temporal and spatial domain. First, non-rigid motion analysis is applied to find the non-rigid motion in the local face regions. The non-rigid motion can be exhibited in the real face while many fake faces cannot. Second, in face-background consistency both the fake face motion and background motion are consistent and dependent. Finally, the banding effect is the only spatial clue that can be detected in the fake images, because the image quality is degraded due to the reproduction. Their techniques show a better generalization capability on different datasets.

In [33]-[34]-[35], the authors analyzed the optical flow in the input image to detect spoofing attacks. The optical flow fields generated by the movement of a two-dimensional object and by a three-dimensional object are utilized to distinguish between real fake face images. They calculated the difference in the pixel intensity of image frames to extract the motion information. The motion information is fed to the classifier to determine whether the input images are real or not. In previous studies, 2D attacks were performed by showing printed photos or videos to the system on a flat surface. However, with the advancement in 3D printing technologies, the detection of a 3D mask against a 2D mask has become more complex and harder to identify [34]. Since the liveness detection and motion analysis fail to detect and protect the system against 3D mask attacks, the texture analysis method is one of the reliable approaches that can detect a 3D mask.

In [36]-[37]-[38], Local Binary Pattern and its variations are proposed to protect face recognition system against 3D mask attacks. As explained before, LBP is used to extract features and generate a histogram using a 3D MAD database. The LBP histogram matching using x2 is applied to compare test samples with a reference histogram. Additionally, both linear (LDA) and non-linear (SVM) classifiers are tested. Principle Component Analysis (PCA) is used to reduce dimensionality, while 99% of the energy is preserved. The Inter Session Variability (ISV), an extension of the Gaussian Mixture Models approach, is applied to estimate more reliable client models by modeling and removing within-client variation using a low-dimensional subspace [39]. Their experimental result shows that using LDA classification is more accurate in 3D mask attacks, especially in the case of a 3DMAD database.

2.2 Open Authorization Protocol

OAuth protocol allows the resource owner to grant permission to a third-party application to access the owner's protected resource, on their behalf, without releasing their credentials (e.g., username, password) [40]. OAuth has provided a mechanism where the third-party application role is separated from the resource owner roles. There are four roles in the OAuth protocol:

- Resource Owner: typically an end user who has the power to grant access to the protected resource [41].

- Resource server: the server that hosts the protected resource owned by the resource owner, and has capability to accept access token.
- Client: a third-party application that issues a request to access the protected resource.
- Authorization server: the server that issues the access token to the third-party application.

Yang and Manoharan [42] give an overview of the vulnerabilities of the OAuth 2.0 protocol. The authors built an attacker model that simulates common network attacks on the OAuth 2.0 protocol that could be carried out to impersonate users, such as the replay attacks module, impersonating attack module, and phishing attack module. The attackers' models were built on two assumptions: 1) the attacker has full access to the network and can eavesdrop on the communication between the three parties (client application, resource server, and authorization server), and 2) the attacker model has unlimited resources to launch an attack. Replay attack module is based on the reuse of an authorization code. The attacker may capture the authorization code and resend it back to the client application to login with the account associated with the specific authorization code. The phishing attack module has shown the misuse of web service to manipulate the user. The attacker can register with any web services to gain a client ID to masquerade as a legitimate website. To launch a phishing attack, DNS cache records are poisoned on the user's machine, therefore, the user is redirected to a malicious client site that he/she did not intend to visit. The impersonating attack module takes advantages of secure vulnerability where the communication between the user-agent and client application is not protected using TLS. The attacker eavesdrops on the authorization code and blocks the original request to prevent the authorization code's use. The attacker initiates a forged request with the client application and send the stolen authorization code to start a new authorization flow.

Leinonen et al. [43] proposed a new mothed to secure OAuth using a portable secure memory card with NFC-enabled service. The authors implemented a software application prototype that shows how the secure credential storage on a user-configurable secure memory card can support an open authentication protocol. Proposed technique provides protection against copying and typing the credentials. The card involves two elements: 1) a consumer key that links the provider to web service, and 2) the user token that links the user to the user's account for the service.

Gomaa [44] et al. provided a novel approach based on web camera to replace fingerprint biometric identifier with finger knuckles utilizing OAuth protocol. The users only need to show knuckles against an integrated web camera in order to capture the knuckles and authenticate instantly. This process requires having three gaps between the fingers visible. On the other hand, in a fingerprint authentication mechanism, users are required to scan their fingers with a certain position in order to allow the system to capture and scan clearly. The web camera is hosted in a cloud web service as a cost-effective acquisition device. The verification process consists of four phases: preprocessing, feature extraction, clustering and retrieving, and classification. The preprocessing phase applies different image filters to improve the picture quality. The feature extraction phase extracts and examines the structural information in the distribution of knuckle minutiae and SURF. The authors used K-Means to cluster and identify groups of samples. In the next stage of the identification process, Naïve Bays validates the identity of the group in the cluster. In the classification phase, Naïve Bayesian network classifies all data into correct class category. The authors did not apply the results on a large scale database and did not cover different case suchs as gaining and skin color.

Meng-Yu et al. designed an API access control architecture based on OAuth protocol. The authors designed a system architecture that consists of three components: the APIs management platform, API provider, and API consumer. The APIs management platform is used to distribute, discover, and consume API. The API provider contacts the APIs management platform to register and publish API information such as capabilities, version, library, etc. After registration, the API Provider gets an API key that can be used to validate the source of the token publication. An API consumer can send a request to the APIs management platform to discover the published APIs and sign a Service Level Agreement. Further, the API provider obtains a client key. The authors are forcing the API consumer to follow the token based Access Control Model. The API consumer requests a temporary token from the APIs management platform, and then presents it to the API provider to validate it with the APIs management platform [3].

2.3 Conclusion

Face liveness detection is an important precursor to online facial recognition. We provide a comprehensive review of the techniques for liveness detection which are categorized into static and dynamic groups. Most static techniques use either texture analysis methods such as Local Binary Pattern operators or Fourier spectra methods such as high frequency descriptor. The texture analysis is more powerful in extracting discriminative features such as the MLBP operator. However, its performance degrades under changing lighting directions and shadowing. The Fourier spectra have the ability to capture high and low frequencies from the input face to detect a spoofing attack. The Difference of Gaussian with Sparse Logistic Regression has achieved a 94% on the NUAA dataset, where the MLBP only achieved a 92%. The Fourier spectra are sensitive to brightness effect, which causes the DoG to fail at detecting the border.

The dynamic techniques are based on the detection of motion over the input frames sequence to differentiate between real and fake faces. Since dynamic techniques utilize more than one frame, dynamic techniques achieve better performance compared with static techniques. Thus, dynamic techniques are slow and difficult to implement. Further, some of the dynamic techniques require users to follow instructions to validate their presence, but not all users may cooperate in this respect. This makes the dynamic methods an unfavorable technique to use in the face liveness methods. There are many different factors that might affect the performance of some of the proposed static and dynamic techniques such as media quality, illuminations and user cooperation. Some studies trained their proposed methods using low quality media, making their technique vulnerable to the use of high-quality media.

CHAPTER 3: FACE LIVENESS DETECTION

3.1 Introduction to biometric authentication

Biometric authentication is an automated method that identifies or verifies a person’s identity based on his/her physiological and/or behavioral characteristics or traits. The biometric authentication method is favored over traditional credentials (username / password) for three reasons: first, the user must be physically present in front of the sensor for it to acquire the data. Second, the user does not need to memorize login credentials. Third, the user is free from carrying any identification such as an access token. An additional advantage of biometric systems is that they are less susceptible to brute force attacks. Biometric authentication can be based on the physiological and/or behavior characteristics of an individual. Physiological characteristics may include, iris, palm print, face, hand geometry, odor, fingerprint, and retina etc. Behaviorial characteristics are related to a user’s behavior: e.g., typing rhythm, voice, and gait.

The ideal biometric characteristics to use in a particular authentication should have five qualities[4]: robustness, distinctiveness, availability, accessibility and acceptability. Robustness refers to the lack of change of a user characteristic over time. Distinctiveness refers to a variation of the data over the population so that an individual can be uniquely identified. Availability indicates that all users possess this trait. Accessibility refers to the ease in acquiring the characteristic using electronic sensors. Acceptability refers to the acceptance of collecting the characteristic from the user. The features that provide these five attributes are then used in a biometric authentication or verification system. Verification is defined as the matching of an individual's information to a stored identity, whereas identification refers to whether an incoming user's data matches any user in the stored dataset. Prior to authentication (verification or identification), an enrollment of allowed individuals is required.

In the enrollment mode, the users are instructed to show their behavioral/physiological characteristics to the sensor. This characteristic data is acquired and passed through one of used algorithms that checks whether the acquired data is real or fake. Moreover, it ensures the quality of the image. The next step is to register the acquired data by performing localization and alignment. The acquired data is processed into a template that is a collection of numbers stored in the database.

In the authentication phase, the biometric system includes four steps before making the final decision: Data Acquisition, Preprocessing, Feature Extraction, and Classification [45] [46].

1) Data acquisition: it is a sensor, such as a fingerprint sensor and web camera, which captures the biometrics data with three different qualities: low, normal, and high-quality.
2) Preprocessing: its duty is to reduce data variation in order to produce a consistent set of data by applying noise filters, smoothing filters, or normalization techniques.

3) Feature extraction: it extracts the relevant information from the acquired data before classifying it.
4) Classification: it is a method that uses the extracted features as input and assigns it to one of the output labels.

The verification mode extracts the relevant information and passes it to the classifier to compare the captured acquired data with the template stored into the database to determine the match[45]. In the identification mode, the acquired data is compared with all users' template in the database to the user [46]-[47]. Figure 3. 1. is a simple description of these three modes.

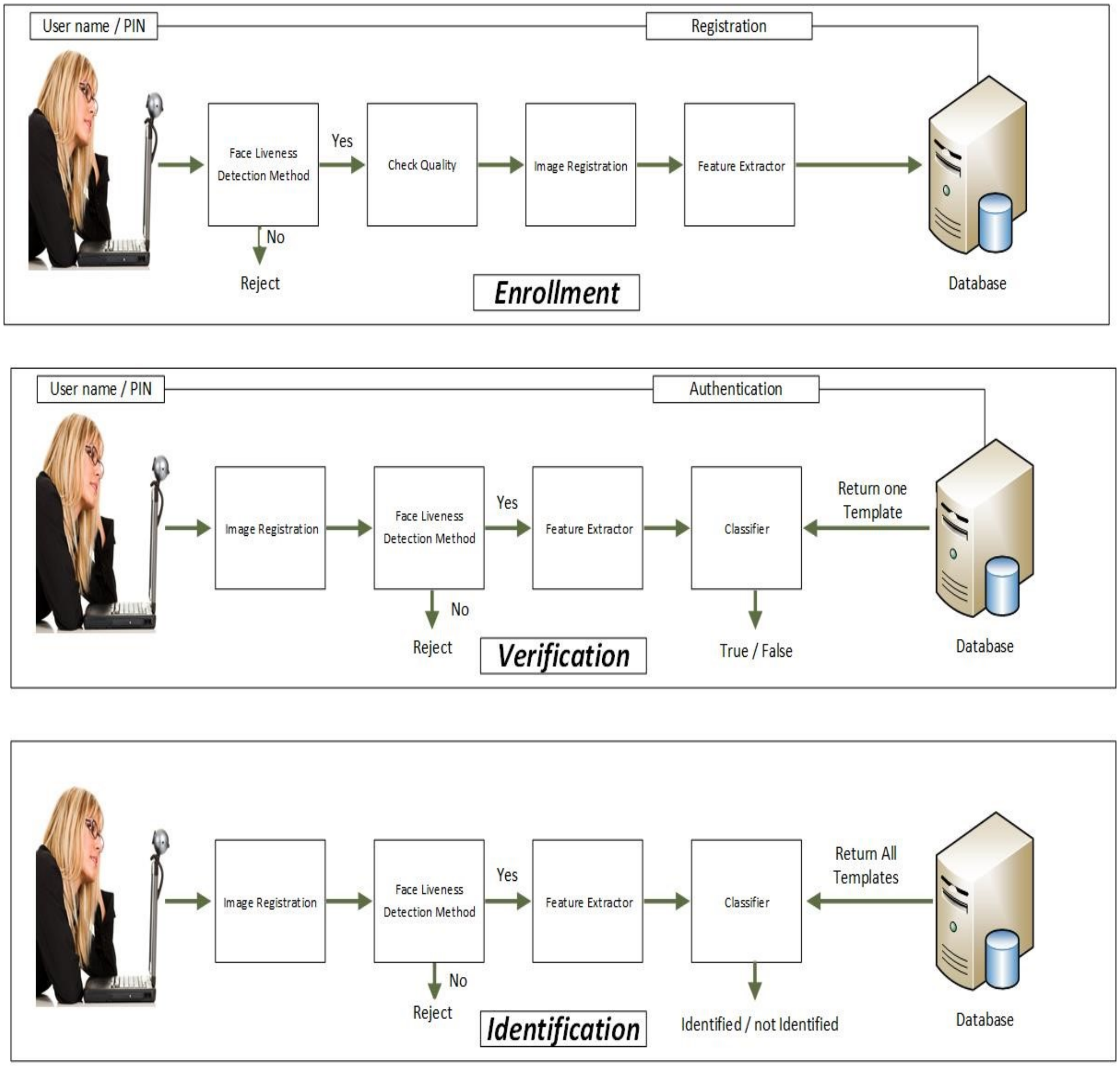


Figure 3.1 Face Recognition System.

For biometric systems based on face recognition, adding a face liveness detection layer to the face recognition system prevents the spoofing attacks. Before proceeding to recognize or verify the user, the face liveness check eliminates the possibility that a picture of the person is presented to the camera instead of the person him/herself.

3.2 Nonlinear Diffusion

The motivation behind our approach is to address spoofing attacks using a nonlinear diffusion based on an AOS scheme with a large time interval to obtain the sharp edges and preserve the boundary locations of the input image. Moreover, we propose a specialized deep convolution neural network architecture that can extract complex and high-level features to distinguish the diffused image, as explained in the next subsection.

3.3.1 Additive Operator Splitting

Nonlinear diffusion is used in our face liveness detection to obtain the sharp edges and preserve the boundary locations that help to distinguish a fake image from a real image by diffusing the input image quickly. Thus, the edges obtained from a flat image will fade out, whereas those from a real face will remain clear. In early computer vision, noise reduction and edge localization in multiscale descriptions of images was developed and explored in the field of image processing [48]. The Gaussian smoothing kernel, a low-pass filter, was used to smooth out image noise—particularly with a multiscale representation using a scale-space parameter (t).

$$I(x, y, t) = I_{Orginal\ Image}(x, y) * G\ (x, y; t). \tag{3.1}$$

Koenderink [49] later noted that convoluting the image with a Gaussian at each scale is equivalent to linear diffusion solutions, such as heat equations.

$$I_t = (I_{xx} + I_{yy}) \tag{3.2}$$

The general diffusion equation is given as

$$\partial I = div(g\ \nabla I) \tag{3.3}$$

Where the diffusivity g is a constant number ∈ ℝ that refers to the speed of the diffusion process. Linear diffusion has some limitations, such as blurring important features, including edges, and dislocating the edges as it smooths a finer scale to a coarser scale [15] [50]. Perona and Malik [51] proposed a nonlinear diffusion method based on a partial differential equation (PDE). They named this approach anisotropic diffusion; this approach avoids the blurring and localization issues that affect linear diffusion as follows:

$$\partial I = \partial_x(g(|\nabla I|)\partial_x) + \partial_y\big(g(|\nabla I|)\partial_y\big) \tag{3.4}$$

Here, the diffusivity g(.) is

$$g(s^2) = \frac{1}{1 + \frac{s^2}{\lambda^2}} \tag{3.5}$$

The nonlinear diffusion filter detects the edges and preserves their locations during the diffusion process using explicit schemes. However, this schema suffers from regularization. Weickert [52] presented a semi-implicit scheme to address this problem. The scheme works with any time step size using an AOS scheme that treats the coordinates of all axes equally, as shown below:

$$(I_k)^{t+1} = \sum_{l=1}^{d} (m\,I - \tau\,d^2\,A_l)^{-1}\,I_k^t \tag{3.6}$$

Where k represents the number of channels and d denotes the input dimension. I is the identity matrix, and A_l is the diffusion in the vertical or horizontal direction. In a case where l = 2 (2D), the equation would be [53] :

$$(I_k)^{t+1} = (2\,I - \tau 4\,A_x)^{-1} I_k^t + \left(2\,I - \tau 4\,A_y\right)^{-1} I_k^t \tag{3.7}$$

The AOS scheme enables fast diffusion even with a large time-step size value (e.g., 100) and can distinguish between edges in flat and rounded surfaces. As shown in Figure 3.2., the edges in printed fake images fade out from the smoothing of the surface texture, whereas the real image preserves its edges and prevents the diffusion from spreading.

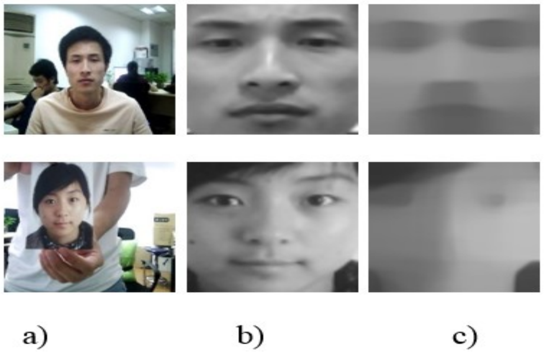
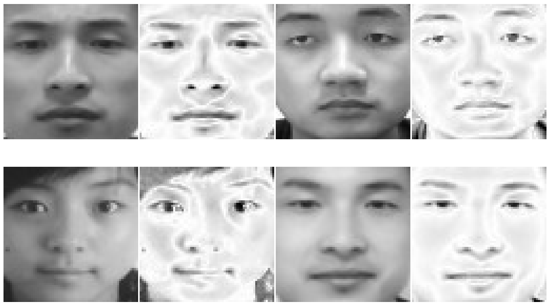


Figure 3.2 a) The top image is a real face; the bottom image is a fake. b) A normalized face with a size of 64 × 64 pixels. c) A diffused image using AOS with a time-step size of 100 and 5 iterations.



a) b) c) d)

Figure 3.3 The top row images (a) and (c) are real faces, whereas the bottom row images (a) and (c) are fake. In both rows, images (b) and (d) represent diffused surfaces scaled from [0, 255].

We extract the information features from the image surface by calculating the diffusion speed as given in [14] [54]:

$$I(x, y) = |\log(I^0\,(x, y) + 1) - \log(I^l(x, y) + 1)| \tag{3.8}$$

Where I^0 represents the original image and I^l denotes the diffused image. As shown in Figure 3.3, the real image surface has relatively sharp edges (e.g., nose and cheek). In contrast, the surface of the fake images have smoother edges. All previous approaches used hand-crafted features, such as the LBP, to extract the information features. That approach has some limitations, such as a limited ability to extract complex features. Therefore, this work uses deep learning with gradient descent to extract the discriminative and higher-level features from the diffused image. We developed a specialized deep convolution neural network architecture to extract the most significant features, which leads to better classification, as explained in the next section.

3.3.2 Convolution Neural network

Machine learning has been successfully applied in many different applications, such as object detection [55], handwriting recognition [56] face detection [57], and face recognition [58]. The convolution neural network (CNN) was first introduced by LeCun et al. [55] [56] and is predominantly a biologically-inspired hierarchical multilayered neural network approach that simulates the human visual cortex and detects translation invariance features [59]. CNNs are designed to extract the local features by

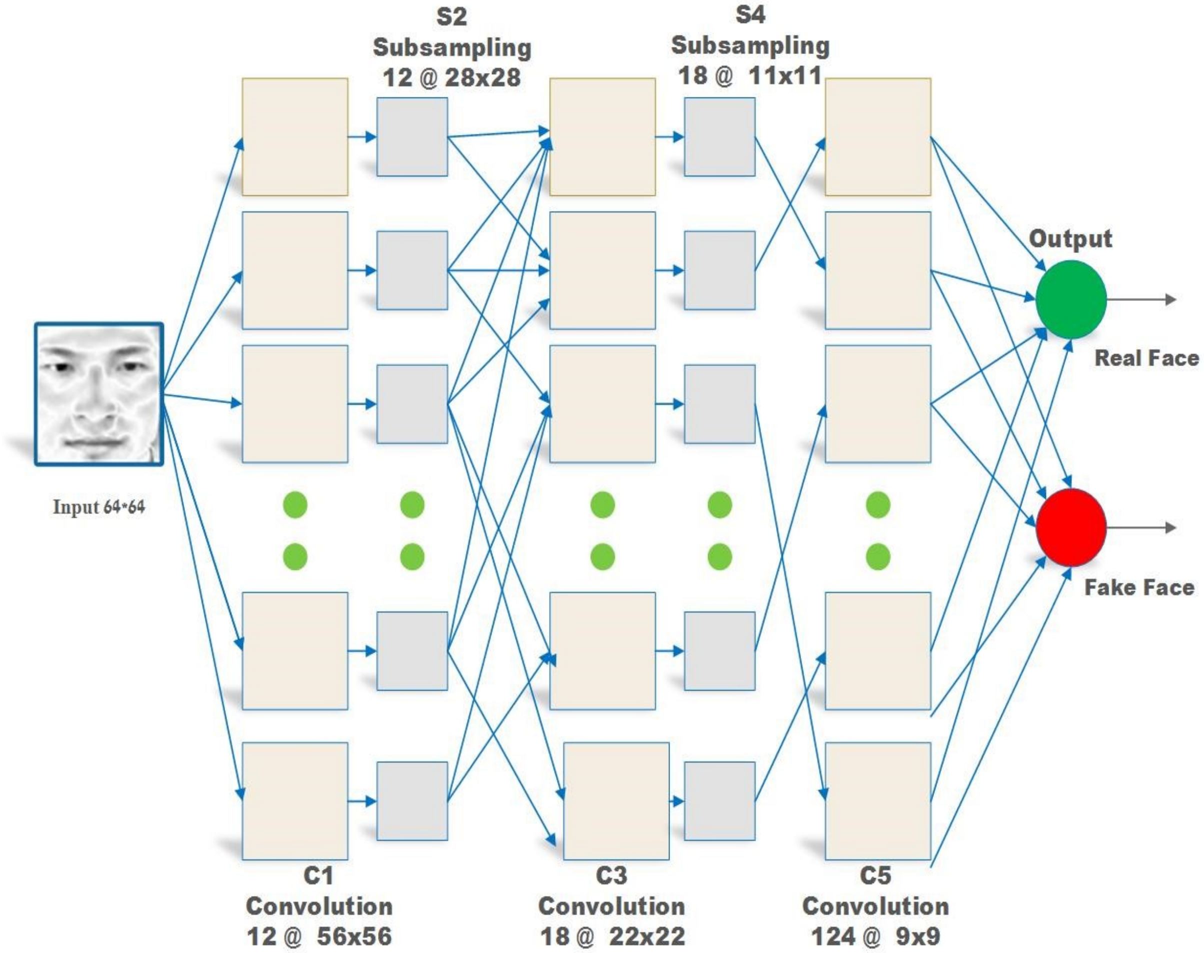


Figure 3.4. Our proposed convolution neural network architecture

combining three architectural concepts that perform some degree of shift, scale, distortion invariance, local receptive fields, shared weight, and subsampling. The ability of both convolution and subsampling layers to learn distinctive features from the diffused image helps in extracting features and achieving the best classification for face liveness detection. Our developed deep convolution consists of six layers. The first five layers are convolutional and subsampling layers, and the last layer is the output layer, as shown in Figure 3.4. The input image, not counted in the CNN layer, has a size of 64 × 64 pixels. Layer C1 is the first convolution layer and consists of twelve feature maps. Each unit in the feature map is a result of connecting a 9 × 9 neighbor in the input image. The new size of the feature map is 56 × 56 pixels. Layer S2 is a subsampling layer with twelve feature maps of 28 × 28 pixels. Each feature map in the subsampling layer is connected to an average kernel 2 × 2 neighborhood from the previous corresponding feature map in C1. The average 2 × 2 kernel is non-overlapping. Therefore, the size of the feature map in S2 is half the size of the feature map in C1. C3 is a convolution layer composed of eighteen feature maps of 22 × 22 pixels. Each feature map takes inputs from four random feature maps from the previous S2 subsampling layer. All four feature maps from subsampling are connected to only one 7 × 7 kernel. Layer S4 is a subsampling layer with eighteen feature maps of 11 × 11 pixels. Each feature map in the subsampling is connected to an average 2 × 2 kernel neighborhood from the previous corresponding feature map in C3. The 2 × 2 kernel is non-overlapping over the input and reduces the subsampling to half the size of the input. C5 consists of 124 feature maps of 9 × 9 pixels. Each unit in the feature map is a result of connecting the 3 × 3 neighbor in the input. Each feature map takes one random feature map from the previous S4 subsampling layer. Finally, the last layer is the output layer, a fully-connected layer. The proposed CNN was trained using the standard backpropagation algorithm, as shown in Table 4.1. We trained our network using the stochastic gradient descent method to calculate the true gradient at each iteration, which is considered faster than batch learning. CNNs learn faster from the unexpected input; thus, we shuffle the data randomly at each iteration. The value of the input image pixels is normalized between zero and one, setting the mean close to zero and the variance close to one. The output of each feature map is normalized or squashed between 1 and -1 using a hyperbolic tangent activation function called Tanh, which helps with the backpropagation learning. All weights (w) and biases (b) are randomly initialized between -1 and 1. We used a small learning rate with a value of 0.005 to help the network learn quickly, especially to update the weight in the backpropagation. The last layer is the fully connected layer; we used the softmax activation function as a classifier to approximate the expected output to be-between 0 and 1 in our binary classification [60].

Table 3.1. Forward and Backpropagation algorithm.

Algorithm 1. Forward and backpropagation algorithms for our proposed convolution neural network

```
All weights (w) and biases (b) are initialized to a value between -1 and 1, and the learning rate λ is set to 0.005

Input (I) of size 64 × 64

For i =1 to I do

Forward

    For layers l=1 to L do

        For FeatureMap f =1 to F do

            If layer l is C layer then

                $\boldsymbol{i_f^{(l)}(x,y) = \emptyset\left((\sum_{k \in K}\sum_{(m,n)} w_f^{(l)}(m,n).\, i_k^{(l-1)}(x+m, y+n)) + b_f^{(l)}(x,y)\right)}$

            Else If layer l is S layer then

                $\boldsymbol{i_f^{(l)}(x,y) = \emptyset\left((w_f^{(l)} * \sum_{(m,n)} i_f^{(l)}(2x+m, 2y+n)) + b_f^{(l)}(x,y)\right)}$

            Else If layer is fully connected then

                $\boldsymbol{i_f^{(l)} = \emptyset\left((\sum_{k=1}^{K}(w_{kf}^{(l)}.\, i_{kf}^{(l-1)})) + b_f^{(l)}\right)}$

            End if

        End for

    End for

Backpropagation

    For layer l=L-1 to 1 do

        If layer l= L then

            $\boldsymbol{\delta_k^{(l)} = (O_k^{(l)} - E_k^{(l)})}$
```

```
      Else if layer l+1 is fully connected then
        $\boldsymbol{\delta}_{k}^{(l)}(x,y) = \left(\sum_{k=1}^{K} \sum_{(x,y)} \boldsymbol{\delta}_{k}^{(l+1)}\ w_{kf}^{(l+1)}(x,y)\right)\ \emptyset'(A_{(k)}^{(l)})$
      Else if layer l+1 is C then
        $\boldsymbol{\delta}_{f}^{(l)}(x,y) = \left(\sum_{k=K_{l\,(Convolution)}} \sum_{(m+i,n+j)} \boldsymbol{\delta}_{k}^{(l+1)}\ (i,j)\ w_{k}^{(l+1)}(m,n)\ \right)\ \emptyset'(A_{(k)}^{(l)})$
      Else if layer l+1 is S then
        $\boldsymbol{\delta}_{f}^{(l)}(x,y) = \left(\ \boldsymbol{\delta}_{k}^{(l+1)}\ (2x+m, 2y+n)/\ w_{f}^{(l+1)}(m,n)\right)\emptyset'(A_{(k)}^{(l)})$
     End if
   End for
   For layer l=1 to L do
     If layer l is C then
        $w_{f}^{(l)}(m,n) = w_{f}^{(old\,)}(m,n) + \left(-\lambda \sum_{k\in K} \sum_{(x,y)} \left(\boldsymbol{\delta}_{f}^{(l)}(x,y)\ i_{k}{}^{(l-1)}(x+m, y+n)\right)\right)$
     Else If layer l is S then
        $w_{f}^{(l)}(m,n) =\ w_{f}^{(l)}(m,n)$
      Else If layer is fully connected, then
        $w_{f}^{(l)}(m,n) =\ w_{f}^{(old\,)}(m,n) + (-\lambda \sum_{k\in K} i_{k}^{(l-1)} . \boldsymbol{\delta}_{f}^{(l)}\ )$
      End if
      If layer is C then
        $b_{f}^{(l)}(x,y) =\ b_{f}^{(old)}(x,y) + (\ -\lambda . \boldsymbol{\delta}_{f}^{(l)}(x,y))$
      Else If layer is S then
        $b_{f}^{(l)}(x,y) =\ b_{f}^{(old)}(x,y) +\ (-\lambda . \boldsymbol{\delta}_{f}^{(l)}(x,y))$
      Else if layer is fully connected then
        $b_{f}^{(l)}(x,y) =\ b_{f}^{(old)}(x,y) +\ (-\lambda . \boldsymbol{\delta}_{f}^{(l)}(x,y))$
      End if
   End for
Until Convergence
```

3.3 Auto-encoder

3.3.1 Introduction

An auto-encoder is conceptually an artificial neural network used for learning useful data representation from unlabeled data through an unsupervised learning method [61]. The auto-encoder consists of an encoder and a decoder that remove the data redundancies and preserve significant data features as shown in Figure 3.5 . The encoder extracts the hidden representation $y_i \in R^{date\,y}$ from the raw data $x_i \in R^{date\,x}$ using a function $g()$:

$$y_i = g(Wx_i) + b$$

Where g() is the sigmoid function and W, x and b represent the weight, the input value and the bias respectively.

The decoder reproduces the initial data $x'_i \in R^{date\,x}$ from the hidden representation $y_i \in R^{date\,y}$:

$$x'_i = g(W'y_i) + b$$

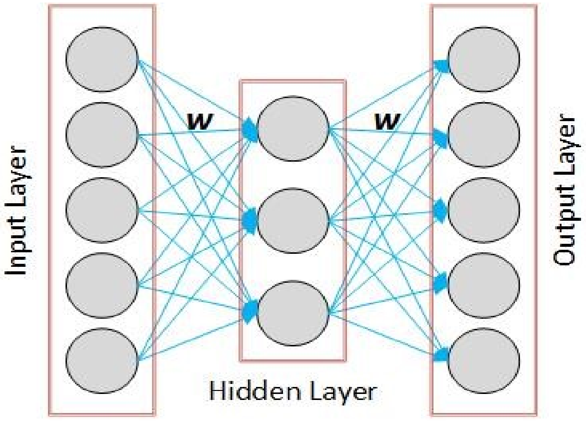


Figure 3.5. Single Layer Autoencoder

3.3.2 Motivation

The successful use of non-linear diffusion followed by deep learning to detect face spoofing attacks in the previous section, motivated us to explore a new method to extract ideal diffused image. We applied the non-linear diffusion based on an additive operator splitting (AOS) equation to extract the diffused image. We tried various numbers of number of iterations and different time steps parameters, which were chosen randomly and tested individually, to obtain the most distinguished diffused image. We proposed a specialized deep convolution neural network to extract complex and higher-level dimensional features. Therefore, we designed a spoofing detection system that utilizes the auto-encoder to automatically discover the ideal nonlinear diffusion's parameters and to extract the complex and high-dimensional features.

3.3.3 Proposed method

We propose a novel multi-model deep learning framework to extract the ideal diffused image along with the significant features to detect the spoofing attack. This multi-model deep learning consists of three stages: The first stage involves an auto-encoder that is pre-trained to produce the diffused image without employing an additive operator splitting (AOS) scheme equation. In the second stage, the output of the auto-encoder is used as input to pre-train the convolution neural network. In the final stage, the entire network both the auto-encoder and the convolution neural network are trained jointly as a single model to adopt the ideal diffused image parameters and weight to enhance the spoofing detection performance.

The auto-encoder is pre-trained to produce a diffused image without applying the nonlinear diffusion question. The input image is the captured web cam image where the output image is a diffused image generated through the non-linear equation setting the number of iteration to 5 and the time step to 100. The auto-encoder architecture consists of three layers. The first layer is the input layer that has 4096 neurons. The second layer is the hidden layer and it is composed of 600 neurons. The last layer is the output layer that has 4096 neurons as shown in Figure 3.6.

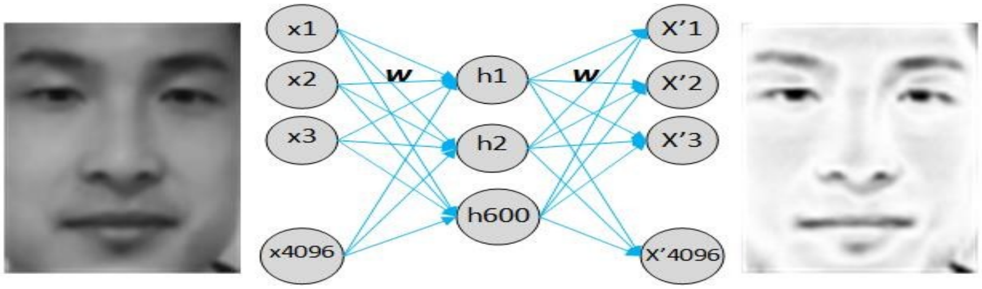


Figure 3.6. Diffused image obtained through auto-encoder

The output of the auto-encoder will be converted to a 2D diffused image. In order to extract the discriminative and higher-level features from the diffused image, a specialized deep convolution neural network architecture is proposed as explained in the next section.

3.3.4 Fine tuning

The entire architecture consists of auto-encoder and convolution neural network which is pre-trained first and then is fine-tuned end-to-end, without hand-craft components to extract the significant features. The entire architecture is trained jointly as single model to adopt the ideal diffused image parameters and the ideal weight to detect the spoofing attack.

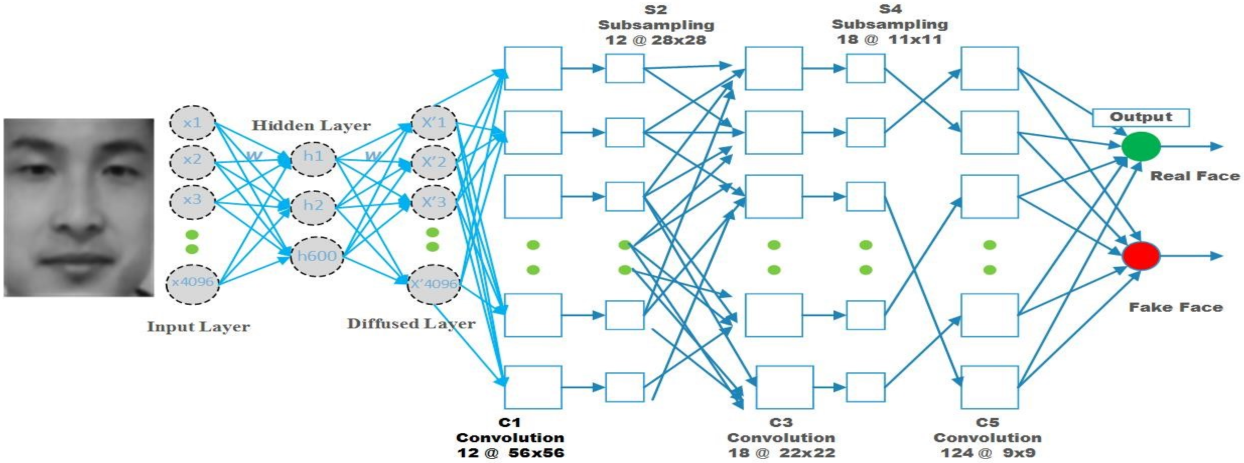


Figure 3.7. Fine tuning

The proposed CNN was pre-trained using the output of the auto-encoder. The standard backpropagation algorithm using the stochastic gradient descent method is used to calculate the true gradient at each iteration, as shown in Table 3.1 . We randomly shuffle the subset data at each iteration which helps the CNNs to learn faster. All weights (w) and biases (b) are randomly initialized between -1 and 1. A small learning rate with a value of 0.005 is used to help the network learn quickly, especially to update the weight in the backpropagation. The output of each feature map is squashed between -1 and 1 using a hyperbolic tangent activation function (Tanh). In the last layer softmax activation function is used as a classifier to approximate the expected output to between 0 and 1 in our binary classification. The input image has a size of 64 × 64 pixels. First convolution layer (C1) consists of 12 feature maps. Each unit in the Convolutional layer is a result of connecting 9 × 9 kernel . Second layer is a subsampling layer (S2) with 12 feature maps. Each feature map in the subsampling layer is connected to an average kernel (2 × 2) neighborhood from the previous corresponding feature map in convolutional layer. Third layer is a convolution layer (C3) composed of 18 feature maps of 22 × 22 pixels. Each feature map takes inputs from 4 random feature maps from the previous subsampling layer. Each unit in the subsampling is connected to 7 × 7 kernel. Fourth layer is a subsampling layer (S4) with 18 feature maps of 11 × 11 pixels. Each unit in the subsampling layer is connected to an average 2 × 2 kernel. Fifth layer is convolutional layer (C5) consists of 124 feature maps. Each unit in C5 is a result of connecting the 3 × 3 neighbor in the input. Each feature map in C5 takes one random feature map from the previous S4 subsampling layer. Finally, the last layer is the fully connected layer, output layer.

3.3.5 Analysis

In this subsection, we discuss and analyze our approach using the autoencoder to detect the face spoofing attacks utilizing a static image. We initially pre-trained the auto-encoder to produce the diffused image, which is similar to the one obtained through Additive Operator Splitting (AOS). When training the entire network jointly, the auto-encoder's weights are updated through the backpropagation algorithm that affects the output of the auto-encoder as shown in Figure 3.7. The auto-encoder is able to extract better depth information and texture surface for real images when compared to the nonlinear diffusion as shown in figure 3.8. In contrast, the nonlinear diffusion obtains more flat surface for spoofing attacks when compared with the auto-encoder. Thus, the trained auto-encoder destroys the edges and texture of some images in the test set, which affects our multi-model deep learning algorithms performance.

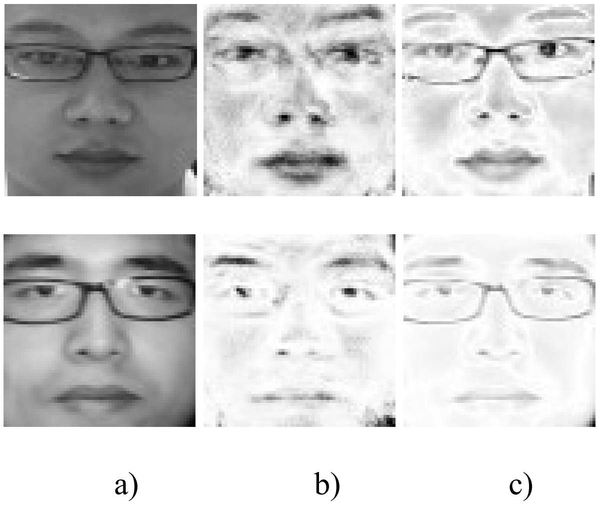


Figure 3.8. Example of the NUAA database, a) The top image is a real face from Test set; the bottom image is a fake face. b) diffused images obtained through the Auto-encoder. c) diffused images obtained through AOS with a time-step size of 100 and 5 iterations.

3.4 Conclusion

In this chapter, an effective and robust approach was proposed to address the problem of face spoofing attacks using a static image. We used an AOS-based schema with a large time-step size to generate the speed-diffused image. The AOS-based scheme was able to detect sharp edges and texture features in the input image. Fake face images had fewer edges and flattened surfaces around the eyes, nose, lips and cheek regions when we recaptured the input image twice, which destroyed the sharp edges and changed the pixel locations. In contrast, real face images had sharp edges and rounded surfaces, especially around the nose and lips. Previous approaches used handcrafted features, such as the LBP algorithm, to extract information features from the diffused image. In contrast, this work used a deep learning algorithm with gradient descent. Our proposed CNN architecture was able to extract the most significant features from the diffused image.

CHAPTER 4: IMPLEMENTATION AND PRELIMINARY RESULTS

4.1 Single image Attack (Picture of a Picture)

5.3.1 NUAA dataset

The NUAA Photograph Imposter Database [18], released in 2010, is publicly available and widely used for evaluating static face liveness detection. The NUAA database consists of 12,614 images of both live and photographed faces that have been captured in three different sessions with an approximately two-week interval between any two sessions. The database images consist of 15 subjects. Specifically, each subject in each session was asked to face the web camera with the goal of capturing a series of facial images with a natural expression and no apparent movement (capturing 20 frames at 20 fps and 500 images for each subject).

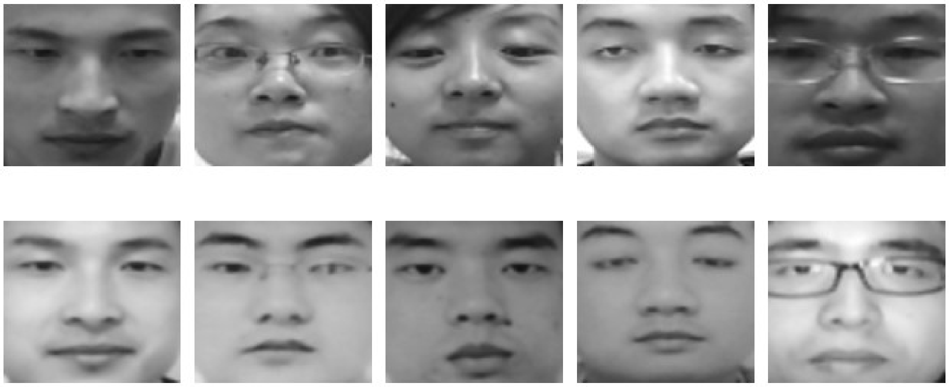


Figure 4.1. Example of the NUAA database (Top: Live photograph, Bottom: Fake photograph)

Table 4.1. NUAA Database

	Training Set			
	Session 1	Session 2	Session 3	Total
Client	889	854	0	1743
Imposter	855	893	0	1748
Total	1744	1747	0	3491
	Test Set			
	Session 1	Session 2	Session 3	Total
Client	0	0	3362	3362
Imposter	0	0	5761	5761
Total	0	0	9123	9123

The database images were converted into a gray-scale representation and resized to 64 × 64 pixels, as shown in Figure 4.1. The database is divided into a training set with a total of 3,491 images and a test set with a total of 9,123 images, as shown in more detail in Table 4.1. The training set consists of images collected from the first and second sessions, whereas the images in the test set were collected only from the third session. There is no overlap between the training and test sets.

5.3.2 Discussion and analysis

In this subsection, we discuss and analyze the efficiency and robustness of our approach for detecting face spoofing attacks utilizing only one input image. The input image is normalized to 64 × 64 pixels. This normalized image has no background, which reduces the time required for processing—particularly when passing the input image through our deep convolution neural network. Some dynamic techniques utilize the background to extract features that increase the time required for processing; however, our approach focuses on extracting sharp edges from the input surface rather than detecting other features from the background. We apply the AOS-based diffusion scheme to extract sharp edges and surface textures, such as the nose, eyes, lips, and cheek. These characteristics form most edges and textures in faces that can help distinguish 2D from 3D images when applying a large time-step value. Re-capturing the input image twice destroys the sharp edges and changes the pixel locations as shown in Figure 4.2.

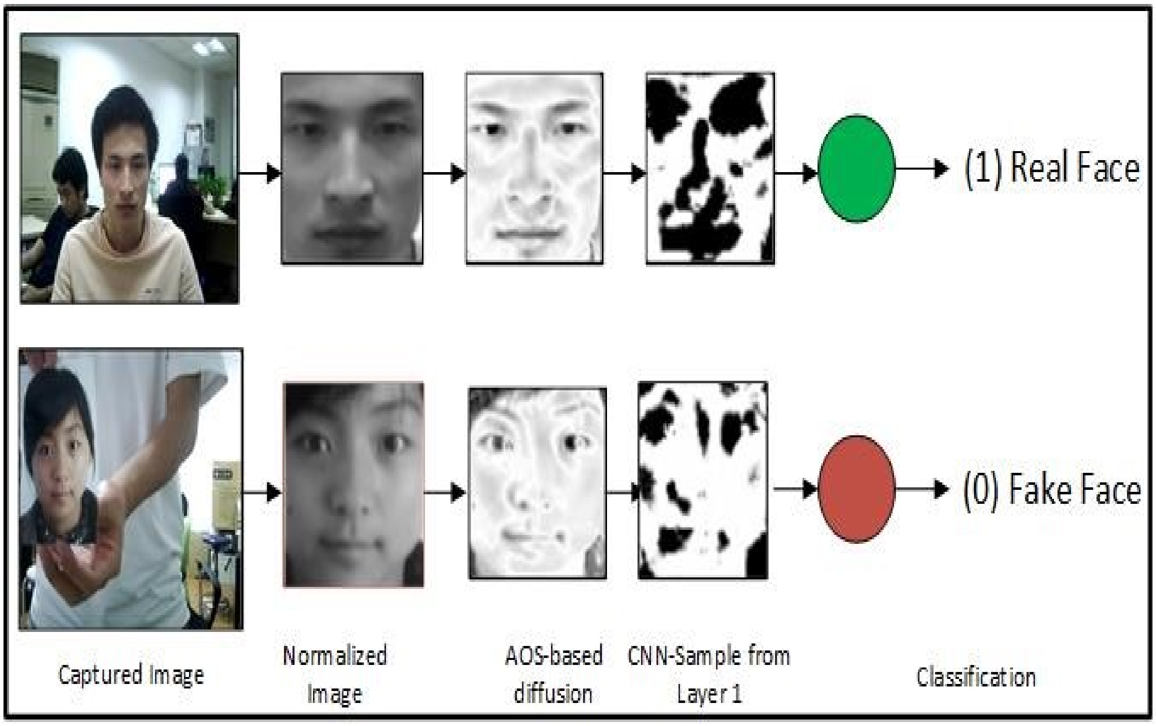


Figure 4.2. Process of our proposed approach.

After conducting many experiments with different time-step values, we determined that a time step of (τ = 100) yields the best result when iterating five times (L =5), as shown in Figure 4.6. Using a larger time step (one greater than τ = 100) causes the most important features, such as the edges and location, to fade out, as shown in Figure 4.3. Applying the AOS-based scheme with a time step of τ = 200 blurs and fades out the edges, thus making it difficult to extract features from either fake or real faces. Moreover, we also tested the impact of the number of iterations on the classification result. We conducted four experiments using four different iterations (5, 10, 15, and 20) while holding the time step constant at τ = 100. Increasing the number of iterations from 5 to 10 blurs the face and consumes additional time, as shown in Figure 4.7. The iteration L = 5 yields an accuracy of 99%, whereas iterations of L = 10 and L = 20 yield accuracy rates of 93% and 92%, respectively. Our proposed feature extraction, CNN, has proven to be powerful in extracting not only the edges but also the textures of the faces as shown in Figure 4.2. The trained kernels are able to detect features that help in distinguishing the speed-diffused images. After visualizing the first convolution layer, there is a clear difference in the real and fake diffused images (e.g., the eye, nose, lips, and cheek regions). The real face has more edges and distinct corners around the eyes and lips, where the fake face has fewer edges and flat surfaces.

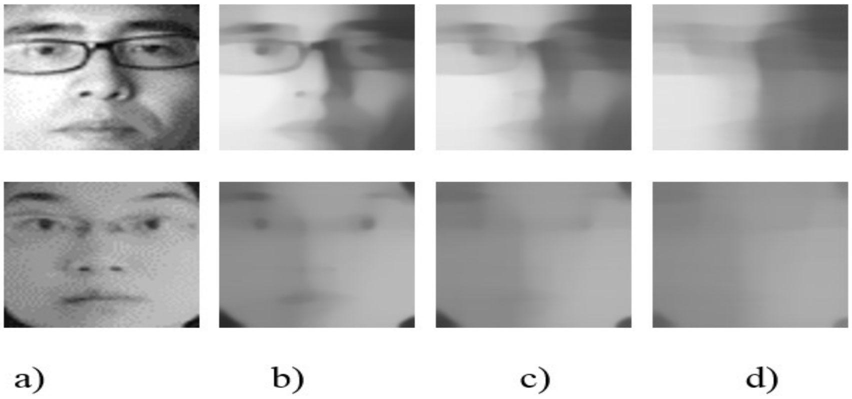


Figure 4. 3. a) Normalized image, b) diffused image with τ = 50, c) diffused image with τ = 100, and d) diffused image with τ = 200

5.3.3 Result

To evaluate the performance of our approach, we conducted many experiments with different time step size values (τ) and different iteration numbers (L) using the images in the NUAA dataset, as shown in Table 4.4. The best detection accuracy achieved using the NUAA dataset was 99% using values of τ = 100 and L=5. Using a larger time step value and a larger number of iterations does not always yield higher accuracy, as shown in Table

4.2. For example, experiments where τ = 120 and L=5 resulted in an accuracy of 98.21%, and experiments where τ = 120 and L=10 resulted in an accuracy of 93.97%. Thus, increasing the iteration number not only fails to improve the accuracy rate of our proposed deep convolution neural network, as shown in Figure 4.7, but also requires more computational time.

Table 4.2. Performance with different parameters using the NUAA dataset.

τ	**L**	**Accuracy**	τ	**L**	**Accuracy**
40	5	90.23	40	10	97.56
60	5	94.03	60	10	93.29
80	5	93.31	80	10	97.36
100	5	**98.99**	100	10	95.99
120	5	98.21	120	10	93.97
140	5	97.96	140	10	94.11
160	5	97.15	160	10	97.01
180	5	96.74	180	10	94.74
200	5	94.10	200	10	94.60

Table 4.3. Performance evaluation for different numbers of iterations. The time step is fixed at a value of 100.

τ	**L**	**Accuracy**	τ	**L**	**Accuracy**
100	1	86.41	100	6	96.30
100	2	90.57	100	7	96.43
100	3	95.20	100	8	95.74
100	4	98.02	100	9	97.00
100	**5**	**98.99**	100	10	95.99

Table 4.4. Performance comparison using the NUAA dataset.

Methods	Accuracy
M-DoG [21]	81.8%
HDF [22]	84.5%
DoG-LRBLR [18]	87.5%
M-LBP [10]	92.7%
DoG-S L [19]	94.5%
CDD [16]	97.7%
DS-LSP [14]	98.5%
Auto-encoder -CNN	97%
Our proposed approach	**99.0%**

To prove the efficiency and effectiveness of our approach, we compared the performance of our proposed deep convolution neural network using the NUAA dataset with all previously proposed approaches, as shown in Figure 4.4. The compared approaches were: multiple difference of Gaussian (M-DoG) [21], high descriptor frequency [22], DoG sparse low-rank bilinear logistic regression (DoG-LRBLR) [18], multiple local binary pattern (M-LBP) [10], DoG-sparse logistic (DoG-SL) [19], component dependent descriptor (CDD) [16], and the diffused speed-local speed pattern (DS-LSP) [14]. As shown in Table 4, our proposed approach achieves the best performance with an accuracy of 99%.

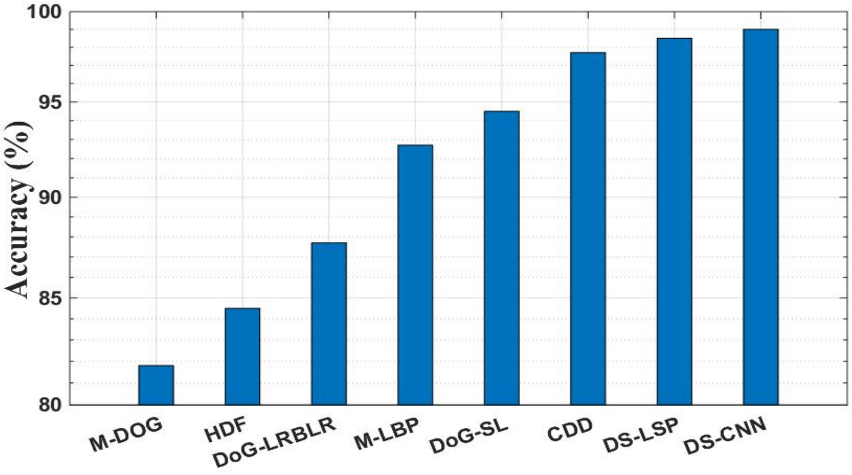


Figure 4.4. Performance comparison using the NUAA dataset.

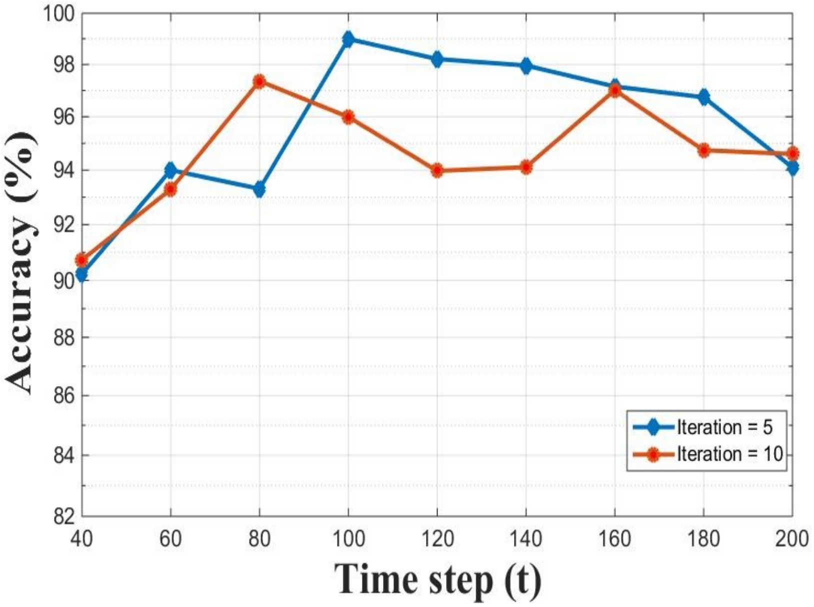


Figure 4.5. Performance comparison between 5 and 10 iterations.

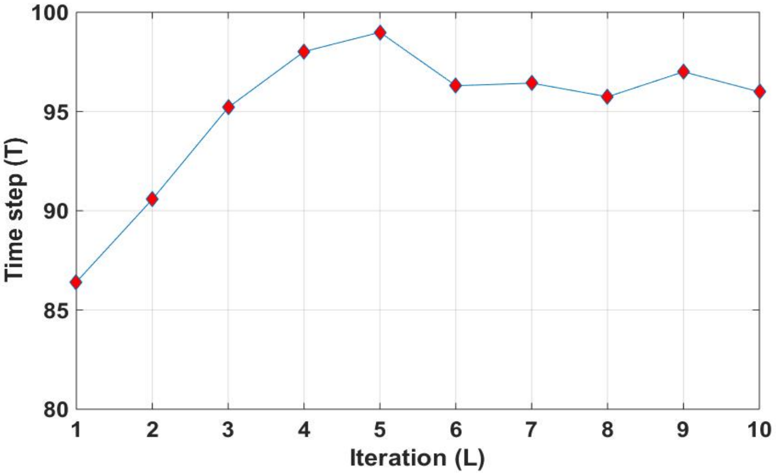


Figure 4.6. Performance evaluation for different numbers of iterations with different time step.

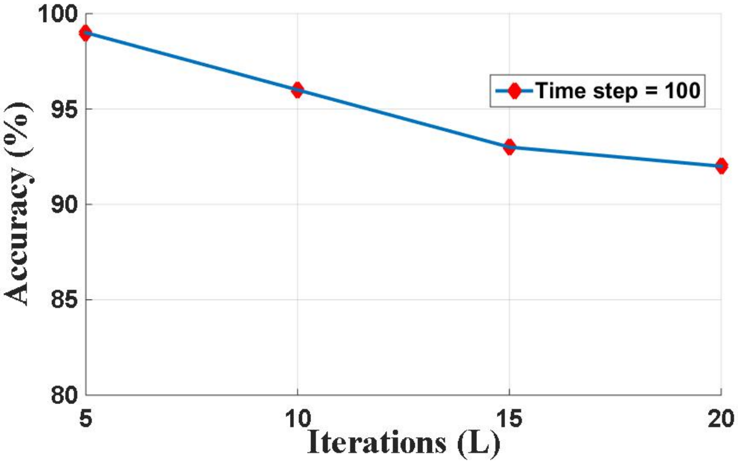


Figure 4.7. Performance evaluation for different numbers of iterations. The time step is fixed at a value of 100.

We computed the half total error rate (HTER) to assess the statistical significance of the performance of our proposed approach [62]. The HTER is half of the sum of the false rejection rate (FRR) and false acceptance rate (FAR), as shown below:

$$\mathbf{HTER} = \frac{\mathbf{FRR} + \mathbf{FAR}}{2}$$

Where FRR is the number of false rejections divided by the total number of clients and FAR is the number of false acceptances divided by the total number of imposters.

Table 4.5. HTER (%) of classification for the NUAA dataset

Method	**HTER**
Our Proposed CNN	0.98%
LBP $_{3*3}^{u2}$ + LDA	17.08%
LBP $_{3*3}^{u2}$ + SVM	19.03%
LBP [10] + SVM	13.17%

Table 4.5 provides a summary of the HTER, showing the classification result of our proposed approach compared with different approaches. The FRR is 0.47%, and the FAR is 1.31%. Our HTER is 0.98%. Analysis of the misclassified face images indicated that over-exposures, blurring, and reflections affected our proposed approach's ability to detect spoofing attacks, as shown in Figure 4.8.

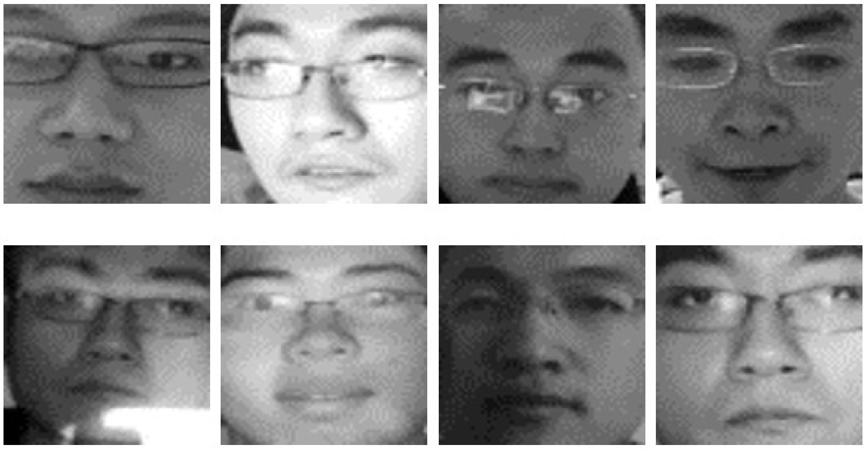


Figure 4.8. Examples of misclassified face images. The top face images are rejected clients, and the bottom face images are accepted printed images.

5.3.4 Statistical Power Analysis

In this section, the power analysis is used to validate the sample size for the experiment and test of the alternative hypothesis. The power of a statistical test can be defined as the probability that the test rejects the null hypothesis (Ho) when the alternative hypothesis is true [73]. Two opposing hypotheses are:

- Null Hypothesis Ho (same, equal, no diff, and no change).
- Alternate Hypothesis Ha (complement of Ha).

The sample size(n) = $\frac{n*\mathrm{p}\ (\mathrm{p}-1)}{(n-1)*(d^2/\ z^2)+p(p-1)}$

Where;

N=12614 , *P*=80%, *d*=5, and *z*=1.96, based on the parameters the sample size (n) = 241

To make the decision of the hypothesis:

Mean $(\overline{X}) = 0.989$, Standard deviation $(\sigma) = 0.01094$, Confidence level (1- α) = 95%,

Critical t (α, d.f) = 1.653, Standard Error (SX) = 0.000704 , t value = 694.6

Both t value and Critical values are used to make the decision. t value > critical t as 694.6 > 1.653 therefore, it rejects Ho and accepts Ha.

5.3.5 Processing Time

In this subsection, we analyze the computational time required by our method in further detail. We divided the processing of our approach to detect spoofing attacks into three steps: the diffusion process, CNN-based feature extraction, and classification. The total time required for the framework of the proposed approach to detect the spoofing attack is approximately 0.076 sec/per person, as shown in Table 4.6. The proposed approach was implemented on a PC with an Intel® Core (TM i7-4500U CPU running at 1.80 GHz and 8 GB RAM without parallel processing. The application was written using Visual Studio 2013 and the C# language. As shown in Table 5.6, the feature extraction using the convolution neural network consumes the bulk of the detection time.

Table 4.6 Time Processing

Diffusion Process	**Our CNN**	**Classification**	**Total**
0.023	0.052	0.01	0.076 (per sec)

4.2 Replay Video Attack

5.4.1 Replay Attack dataset

Replay-Attack Database [13] consists of 1200 short videos of both real-access and spoofing attacks of 50 different subjects. Each person recorded a number of videos with a resolution of 320 x 240 pixels under two different conditions: (1) the controlled condition contained a uniform background fluorescence lamp illumination; and (2) the adverse condition contained a non-uniform background and day-light illumination. The spoof attacks were generated by using one of the following scenarios: (1) a hard copy of photograph, (2) on a phone using an iPhone screen, and (3) on a tablet using an iPad screen. Each spoof attack video was captured in two different attack modes: hand-based attacks and fixed support attacks [32]. The Replay-Attack database is divided into three subsets: training, development, and testing.

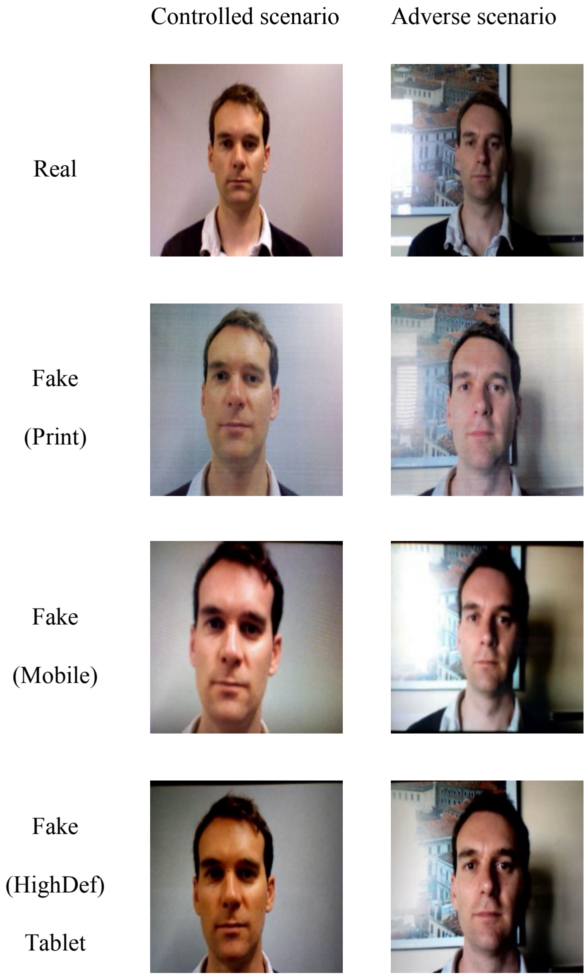


Figure 4. 9. Examples of Replay Attack Dataset (Controlled and Adverse scenario)

Table 4.7. Replay-Attack Database

Type	Training Fixed |hand	Development Fixed | hand	Test Fixed | hand	Total
Genuine face	60	60	80	200
Print-attack	30 + 30	30 + 30	40 + 40	100 + 100
Phone-attack	60 + 60	60 + 60	80 + 80	200 + 200
Tablet-attack	60 + 60	60 + 60	80 + 80	200 + 200
Total	360	360	480	1200

5.4.2 Implementation

Our proposed method utilizes only one frame from sequenced frames of replayed video attacks to detect a spoofing attack. The single frame is captured from different medium such as print photographs, mobile screens and high definition (Tablet) screens that requires less processing times. Due to the recent success reported in [14], we applied a non-linear diffusion to extract shape edges and corners contained in the input frames. Then we used a specialized deep convolution network to detect the texture surface and the edges in order to distinguish a fake image from a real image. We applied nonlinear diffusion to obtain the sharp edges and preserve the boundary locations unlike linear diffusion which blurs important features, including edges, and dislocates the edges as it smooths a finer scale to a coarser scale [15] [50].

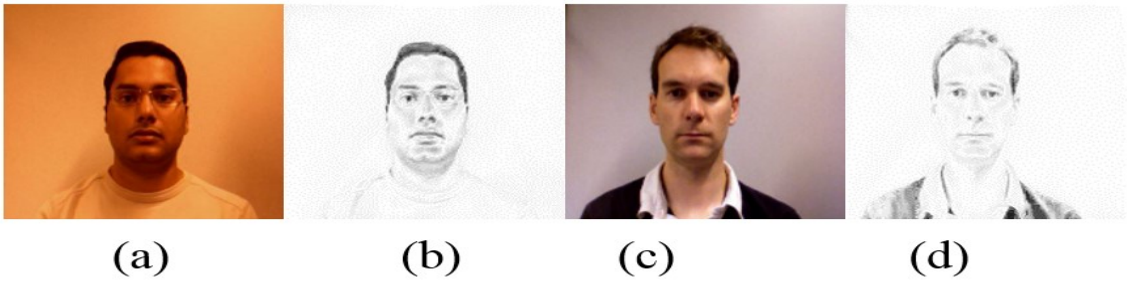


Figure 4.10 . (a) and (b) Original face of real access – Controlled scenario (c) (d) Diffusion speed map scaled from [0, 255].

Our proposed deep convolution network consists of six layers. The first five layers are convolutional and subsampling layers, and the last layer is the output layer. The input image, not counted in the CNN layer, has a size of 320 × 240 pixels. Layer C1 is the first convolution layer and consists of twelve feature maps. Each unit in the feature map is a result of connecting a 9 × 9 neighbor in the input image. The new size of the feature map is 312 × 232 pixels. Layer S2 is a subsampling layer with twelve feature maps of 156 × 116 pixels. Each feature map in the subsampling layer is connected to an average kernel 2 × 2 neighborhood from the previous corresponding feature map in C1. The average 2 × 2 kernel is non-overlapping. Therefore, the size of the feature map in S2 is half the size of the feature map in C1. C3 is a convolution layer composed of eighteen feature maps of 148 × 108 pixels. Each feature map takes inputs from two random feature maps from the previous S2 subsampling layer. All two feature maps from subsampling are connected to only one 9 × 9 kernel. Layer S4 is a subsampling layer with eighteen feature maps of 74 × 54 pixels. Each feature map in the subsampling is connected to an average 2 × 2 kernel neighborhood from the previous corresponding feature map in C3. The 2 × 2 kernel is non-overlapping over the input and reduces the subsampling to half the size of the input. C5 consists of 124 feature maps of 66 × 46 pixels. Each unit in the feature map is a result of connecting the 9 × 9 neighbor in the input. Each feature map takes two random feature maps from the previous S4 subsampling layer. Finally, the last layer is the output layer, a fully-connected layer. The output of each feature map is normalized or squashed between 1 and -1 using a hyperbolic tangent activation function called Tanh, which helps with backpropagation learning. All weights (w) and biases (b) are randomly initialized between -1 and 1. We used a small learning rate with a value of 0.005 to help the network learn quickly, especially to update the weight in the backpropagation. The last layer is the fully connected layer; we used the softmax activation function as a classifier.

5.4.3 Discussion and analysis

In this subsection, we discuss and analyze our approach for detecting replay video attacks utilizing only one frame of a single video as shown in figure 4. 11. Utilizing one frame instead of 375 frames from a single video reduces the time required for processing which in the users' experience is convenient. When we apply the AOS-based diffusion scheme to obtain sharp edges and surface textures, such as the nose, eyes, lips, and cheek, we found that the real access frame and the high definition (Tablet) frame have similar edges and texture which makes it hard for our specialized convolution neural network to distinguish between them. Re-capturing the replay video twice destroys the sharp edges and changes the pixel locations. After conducting several experiments with different time step values, we determined that a time step of (τ = 100) yields the best result when iterating five times (L =5), as shown in Table 4.9. Using a larger time step (one greater than τ = 100) fades out the most important features, such as the edges and location. Moreover, we also tested the impact of the number of iterations on the classification result. We conducted four experiments using four different iterations (5, 10, 15, and 20) while holding the time step constant at τ = 100. Increasing the number of iterations from 5 to 10 blurs the face and consumes additional time. The iteration L = 5 yields a HTER of 10%, whereas iterations of L = 10 and L = 20 yield accuracy rates of 14.625% and 17.375%, respectively as shown in Figure 4.10. Our proposed specialized CNN has proven to be powerful in extracting not only the sharp edges but also the texture information of a single frame. The trained kernels are able to detect features that help in distinguishing the speed-diffused frame. After visualizing the first convolution layer, there is a clear difference in the real and fake diffused frames (e.g., in the eye, nose, lips, and cheek regions). The real face has more edges and distinct corners around the eyes and lips, where the fake face has fewer edges and flat surfaces [63].

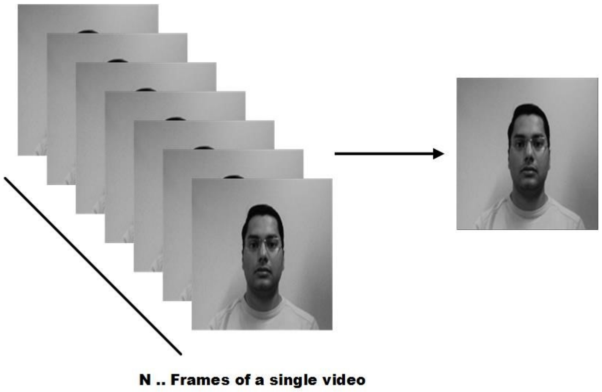


Figure 4.11 . One frame of a short video of real access.

5.4.4 Result

We computed the half total error rate (HTER) to measure the performance of our proposed approach [62].

Table 4.8 . HTER (%) of classification for the Replay Attack dataset

Algorithms	test
LBP $_{3*3}$ ^{u2} + x^2	34.01%
LBP $_{3*3}$ ^{u2} + LDA	17.17%
LBP $_{3*3}$ ^{u2} + SVM	15.16%
DS-Local Speed Pattern	12.50%
Our proposed approach	**10%**

Table 4.9. HTER (%) of classification with different parameters using the Replay Attack dataset

τ	**L**	**Accuracy**	τ	**L**	**Accuracy**
40	**5**	**17.125**	**100**	**5**	**10.00**
60	**5**	**13.125**	**120**	**5**	**15.875**
80	**5**	**13.75**	**140**	**5**	**14.625**

Table 4.10. Performance evaluation for different numbers of iterations. The time step is fixed at a value of 100.

τ	L	Accuracy	τ	L	Accuracy
100	**1**	**16.5**	**100**	**6**	**11.875**
100	**2**	**13.5**	**100**	**7**	**11.875**
100	**3**	**15.75**	**100**	**8**	**12.625**
100	**4**	**14.875**	**100**	**9**	**11**
100	**5**	**10.00**	**100**	**10**	**14.625**

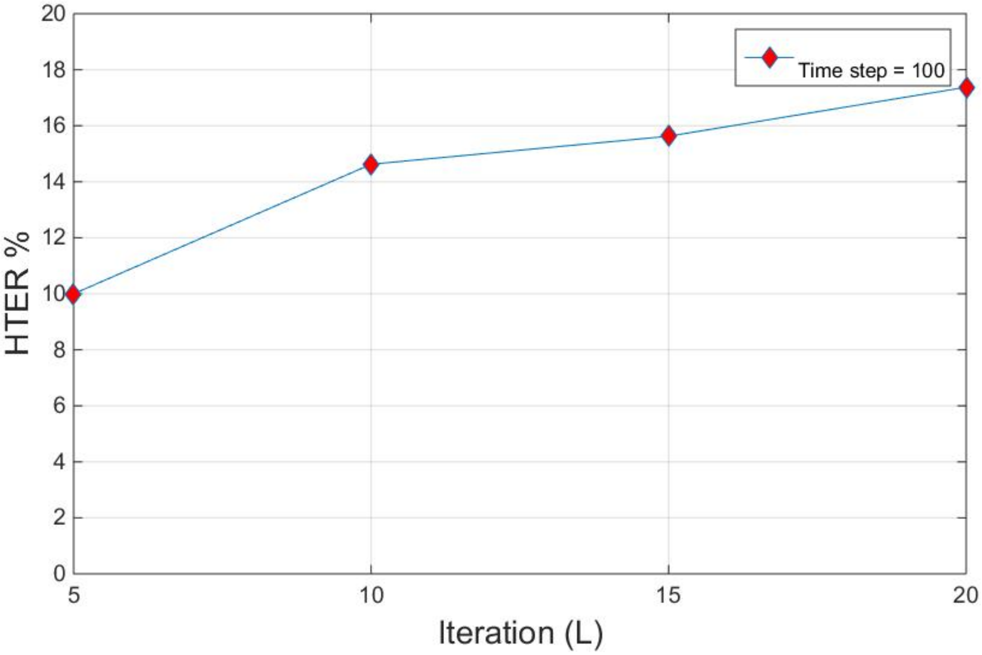


Figure 4. 12 Performance evaluation for different numbers of iterations. The time step is fixed at a value of 100.

CHAPTER 5: ENHANCING OAUTH SECURITY USING FACE RECOGNITION

6.3 Introduction to Open Authorization Protocol

In the traditional authentication mechanism, the client provides his/her credentials to the host server to access its protected resource. With the growing number of third-party applications seek to access the end-user resource, the user shares his/her credentials (username, password) with the third-party. The third-party application knows the credentials, and thus may obtain an overly-broad access to the protected resource [3]. For instance, a third-party web application using Google sign-in can access other google APIs and create entries to the google user's calendar[42]. The main issues with sharing the credentials are [64]:

- The third-party application owns the owner's resource credentials, and has broad access to the online resource with no restrictions.
- A resource owner may revoke access from third-party applications only by changing his/her credentials, so all third-party applications cannot access the resource by using pervious credentials.

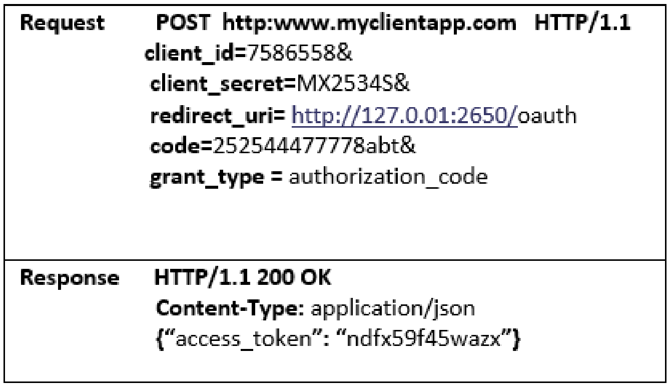
To solve the above problems, Open Authorization (OAuth) protocol has been proposed. OAuth does not require the resource owners to reveal their credentials with third-party applications; instead the third-party obtains an access token issued by the authorization server with the approval of the resource owner[65]. An access token is a string that denotes a specific scope of permission and a duration of lifetime access. The OAuth 2.0 has not defined how the access token should be structured, and how it should be validated: OAuth 2.0 has left that up to the authorization and resource server. The access token can be created as: a Security Assertion Markup Language (SAML) token, Simple Web Token (SWT), or JSON Web Token (JWT).

OAuth 2.0 specifications have defined four authorization grant flows: authorization code grant flows, implicit grant flows, resource owner password grant flows, and client credentials grant flows. A protected resource hosted by the server uses the authorization code grant, while a user agent-based application such as JavaScript uses implicit grant flows [7]. This dissertation focuses only on authorization code grant flows that requires two major steps to grant access:

- After the third-party application initiates the request by redirecting the resource owner to the authorization server to grant the scope of permissions, the authorization server authenticates the resource owner and returns an authorization code to the end-user.
- The third-party application exchanges the authorization code and the client secret for the access token with the authorization server.

As depicted in Figure 5. 3, OAuth specifications are trying to prevent the access token from leaking into the browser, and into the resource owner. Therefore, for exchanging the authorization code for an access token, both the client ID and Client secret are required as depicted in Figure 5.1. Since the resource owner does not know the client secret, he/she cannot obtain the access token.

Figure 5. 1. shows a POST request in the HTTP header request to exchange the code for the access token:



```
Request      POST  http:www.myclientapp.com   HTTP/1.1
            client_id=7586558&
             client_secret=MX2534S&
             redirect_uri= http://127.0.01:2650/oauth
             code=252544477778abt&
             grant_type = authorization_code

Response     HTTP/1.1 200 OK
             Content-Type: application/json
             {“access_token”: “ndfx59f45wazx”}
```

Figure 5. 1. Access token request

6.4 Introduction to Biometrics System

Biometric authentication is the automatic authentication that identifies a person by analyzing their physiological and/or behavioral features. Both physiological and behavioral characteristics have been used in biometric authentication systems for the past twenty years. Physiological characteristics are related to the shape of the person’s body: hand geometry, palm print, face recognition, DNA, iris recognition, retina, fingerprint, iris recognition, and odor. Behavioral characteristics are related to the person’s behavior: typing rhythm, gait, and voice [46]. Using the biometric authentication method to verify the identity of a user is preferred over traditional authentication (user / password) for different reasons: firstly, the user is required to be physically present. Secondly, the user is not required to remember credentials or carry an access token.

The selection of best specific biometric characteristics for a particular application comprises of five qualities: robustness, distinctiveness, availability, accessibility, and acceptability. Robustness means this individual trait will not change over time. Distinctiveness means having a variation over the population. Availability means the whole population should ideally possess this measure in multiples. Accessibility means it is easy to observe the trait using electronic sensors. Acceptability means people do not have an issue with this measurement being taken [66].

All biometric authentication systems follow the same procedures to authenticate individuals: the first step is called enrollment, where the new user is registered into a database. After that, one of behavioral/physiological characteristics of the person is captured and passed through one of the used algorithms that turns the data into a template. The template is a collection of numbers that represents the original biometrics into the database. In order to recognize the person, new measurements need to be captured and translated into a template using the same algorithm that the original template was passed through, and then the new template needs to be compared with the stored template to determine if they match or not [46]. Figure 5. 2 shows the biometric authentication process:

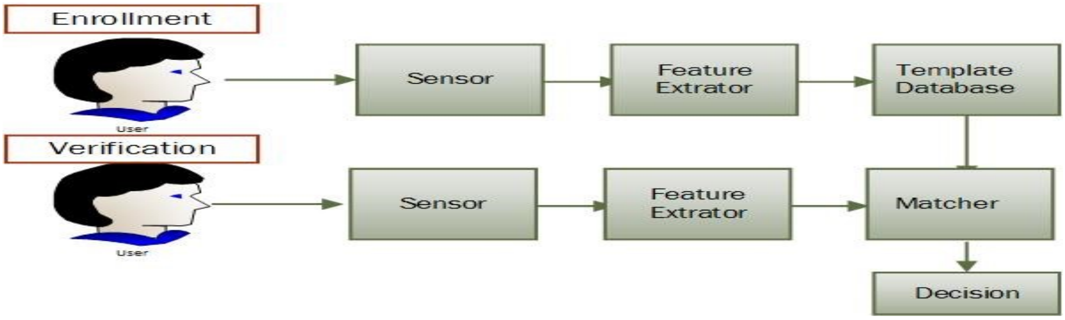


Figure 5. 2. Biometric authentication system

6.5 Login Services

We presented a Secure Login Service using face recognition that authenticates the identity of the user. We created four services to verify the user, and issue the access token; an image registration service, verification service, and an access token service in the authorization server [47]. In OAuth 2.0, the authorization server uses the traditional credentials mechanism to authenticate the resource's owner as shown in Figure 5. 3, step 3.

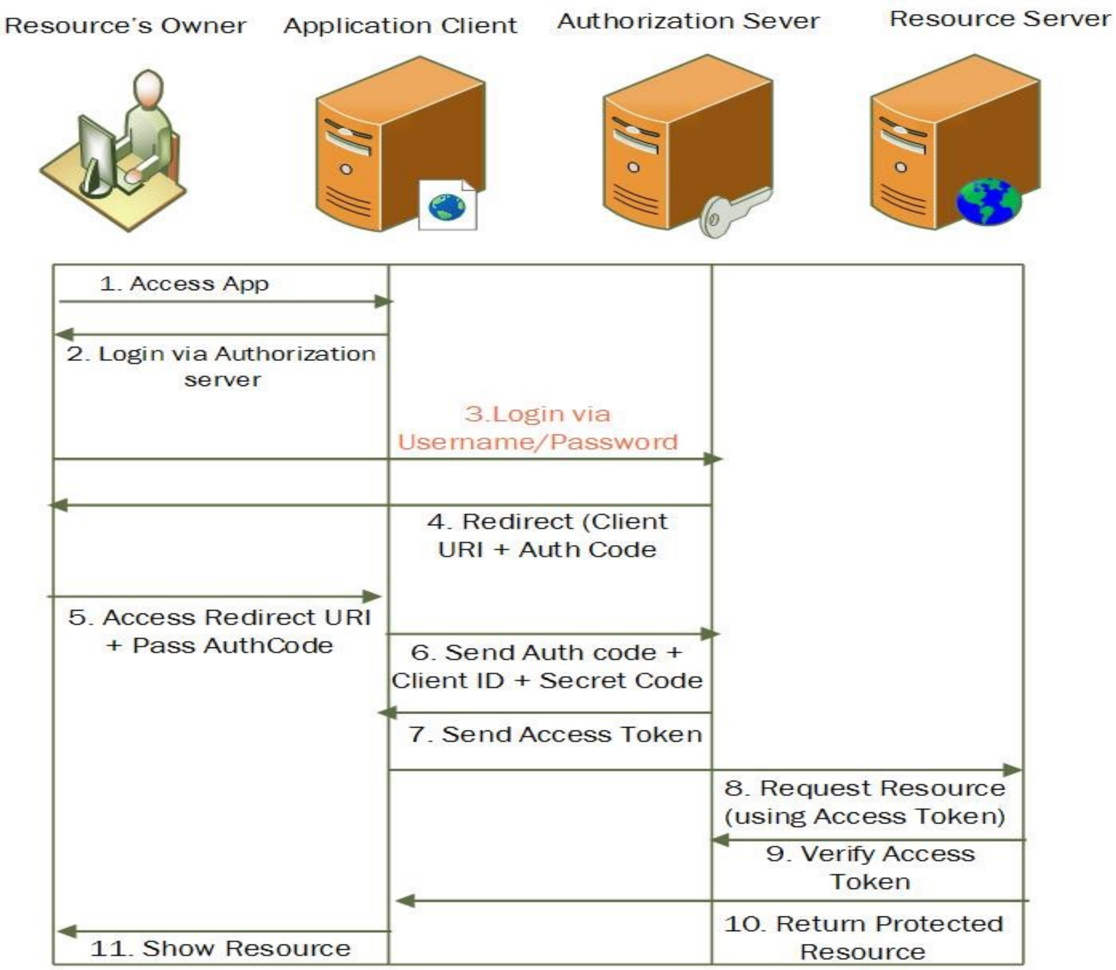

Resource's Owner
Application Client
Authorization Sever
Resource Server
1. Access App
2. Login via Authorization server
3.Login via Username/Password
4. Redirect (Client URI + Auth Code
5. Access Redirect URI + Pass AuthCode
6. Send Auth code + Client ID + Secret Code
7. Send Access Token
8. Request Resource (using Access Token)
9. Verify Access Token
10. Return Protected Resource
11. Show Resource

Figure 5. 3. OAuth 2.0 using Authorization code

Our approach has four services to authenticate the user and issue the access token as shown below:

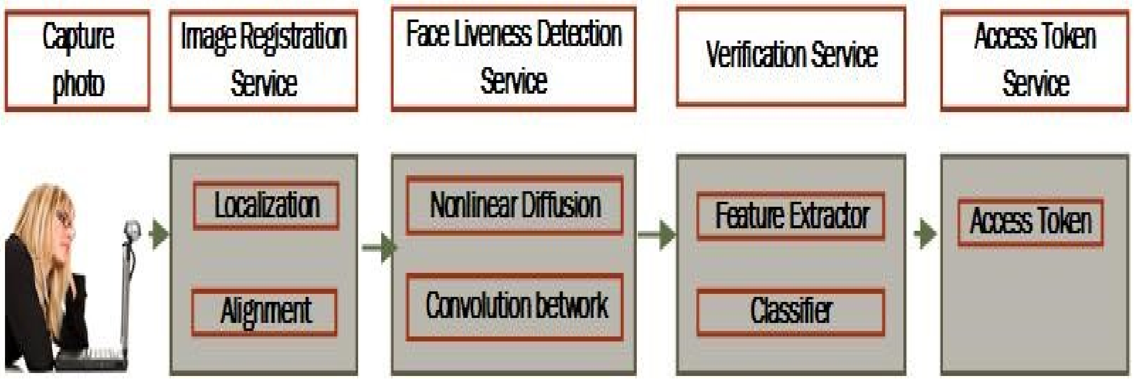


Figure 5. 4. Our proposed Authentication Service

6.5.1 Image registration service:

The main goal of the image registration service is to resize, shift, and rotate the incoming image to reflect the same image size in the database: we assume that when the user's account has been created, the user's identification image has been processed by Image registration service. After capturing the photo and converting it to a one dimensional array [255, 23, 25, …., n], the array passes through the image registration service. Since the user might capture the image from different distance and different alignments, preprocessing is needed to perform a face recognition.

Two preprocessing occurs prior to the authentication of the user. The first preprocess locates landmarks in both images (the captured image and the referenced image in the database) using the active shape model algorithm. The second preprocess computes the transformation.

The Active Shape Model (ASM) Algorithm is a statistical model that represents the shape as landmarks points. We use ASM to locate the five landmarks in the face: the outer right eye, center right eye, outer right eye, center left eye, and the center of the nose.

Once the ASM is applied, the nose landmark coordination in the captured image is (113, 173), and the referenced image is (49, 93). Transformation (T) will be applied to the captured image (I1), so the difference between (x1, y1) in the referenced image (Io) and the captured image (x2, y2) is as small as possible.

$$\begin{pmatrix} X_1 \\ Y_1 \end{pmatrix} = \begin{pmatrix} a & -b \\ -c & d \end{pmatrix} \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} + \begin{bmatrix} t_2 \\ t_2 \end{bmatrix}$$

Where;

- a,b, rotate the image
- b and c resize the image
- t1 and t2 shift the image

To optimize the Cost = (Io – T(I1))2.

C = (x1-(ax2 + by2)+ t1)2 + (y1-(-bx2+ay2)+t2)2.

In order to find the rotation for the five landmarks, we find partial derivatives respect to ∂c/∂a=0.

$\underline{\partial}c/\underline{\partial}a = a(2x_2^2+2y_2^2) + 0 + (-2x_2)+ (-2y_2) = 2x_1x_2 + 2y_1y_2.$

$\underline{\partial}c/\underline{\partial}b = b(0) + (2x_2^2+2y_2^2)+(- 2y_2^2)+ (-2x_2) = 2x_1y_2 – 2x_2y_1$

$\partial c/\partial t1 = t1(2x_2) + (2y_2) + (-2) + 0 = -2x_1$
$\partial c/\partial t2 = t2(2y_2) + (-2x_2) + 0 + (-2) = -2y_1$

The five landmarks of captured and referenced images are calculated to find a, b, t1, t2 values using the transformation method. Then we need to update the new location (x2, y2) for the captured image using:

$$L(\,x_2, y_2) = \sum_{x1,y1}^{0} (a * x_1 + b * y_1 + t1\,, (-b) * \, x_1 + a * \, y_1 + t2 \qquad (5.1)$$

6.5.2 Face liveness detection service

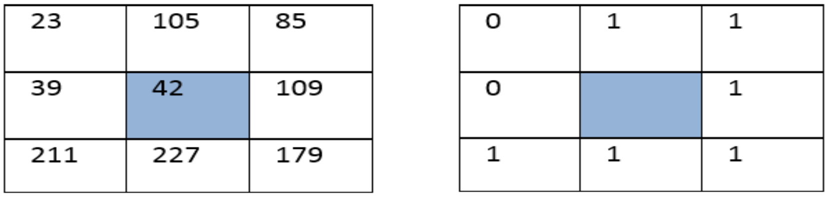
Face liveness detection service is used to prevent the spoofing attacks. We applied nonlinear diffusion followed by a specialized convolution neural network to detect the liveness of the input image. The steps to detect the spoofing attack are explained in chapter 3.

6.5.3 Verification service

Verification service is an applied method that verifies or denies the user from the registered image using the Local Binary Pattern (LBP), which is a nonparametric method. This service consists of feature extraction and classification. LBP is used to extract features by summarizing local structures of the registered image.

LBP Methodology:

The registered image is converted to gray- scale and then divided into a 5 by 5 block. Each pixel in the block is compared with its eight neighborhoods.



23	105	85
39	42	109
211	227	179

0	1	1
0		1
1	1	1

Figure 5. 5. Example of basic LBP codes

$$LBP_{Point,Radius}(x_c + y_c) = \sum_{P=0}^{P-1} S(i_p - i_c)2^p \tag{5.2}$$

Where;

- xc,yc represent the center pixel
- p represents the surrounding pixel
- s(z) = $\begin{cases} 1, & if\ z \geq 0 \\ 0, & if\ z < 0 \end{cases}$

The new value for the center pixel is calculated by concatenating all new binary values in a clockwise direction. The generated binary number is 01111110, equals to 126 as a decimal value [67].

So the center value will be updated with the new value 126, and iterate through the entire block updating each pixel. LBP uses a uniform pattern to describe the texture image. If the generated binary number contains at most two bitwise 0 -1 or vice versa transaction, then the LBP is called uniform. For instance, (01111110), (1100 0000), and (0001 1000) are uniform where (0101 000), (0001 0010), and (0100 0100) are non-uniform. There are 58 uniform LBP Pattern and 198 non-uniform LBP patterns. Each region or block uses one histogram with 58 uniform patterns to provide a distribution description of the local block.

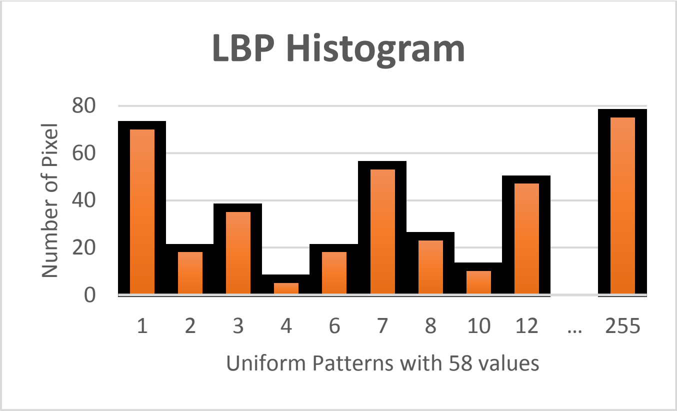


Figure 5. 6. LBP Histogram for one block.

The registered image has 1450 features: 5 * 5 * 58. After calculating the features of the registered image, we apply the LBP uniform patterns to the user's identification image in the database, and obtain 1450 features. Both features are stored in one vector array and passed through a correlation method to find the relationship.

$$R(x,y) = \frac{\sum(\mathrm{x}-\bar{x})(y-\bar{y})}{\sqrt{\sum(}(\mathrm{x}-\bar{x})^{2(y-\bar{y})^2}} \tag{3.3}$$

If the correlation result shows an adequate percentage, the verification service will verify the resource owner.

6.5.4 Access token service

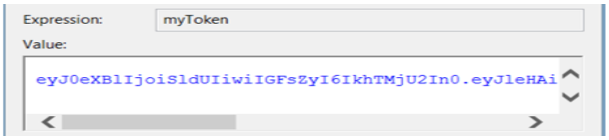
The access token service is responsible for issuing a signed access token as JSON Web Token (JWT) with a thirty second duration and a specific scope of permission. The access token is made up of name-value pairs based on JSON format. For instance,

["expiration": "1427131712", "role": "Admin" , ..].

The access token is transmitted using HTTP over TLS. Also, it is protected from being tampered with using the HMAC-SHA 256 algorithm which combines a hash message with a key. HMAC-SHA 256 name-value pairs are used to ensure the integrity and authenticity of the token. Once the verification service returns with true/verified, the access token service issue the access token and sends it to the user (requester) in Base64 URL encoded format. The resource owner has to send the access token to the third-party application before the expiration duration.

6.5.5 Implementation

We have designed and developed a real-time software application that implements OAuth 2.0. This login service authenticates the identity of the user through a web camera, and issues an access token using HTML 5 and web API technologies. All four services are based on Representational State Transfer (REST) architecture style. The resource's owner will be redirected to a web page that captures an image as shown in Figure 5.3 by using HTML 5 technology, which then convert the image into a one-dimensional array. The registration service uses the ASM algorithm to locate the targeted points in the captured image, and then retrieves the referenced image from the database. We computed a, b, t1, t2 values by using the transformation method. Image registration performs the alignment. After the user is verified, the access token is issued as shown in Figure 5.7:



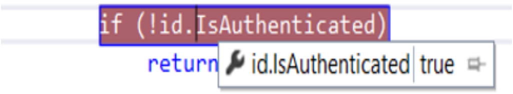
Expression: myToken

Value:

```
eyJ0eXBlIjoiSldUIiwiIGFsZyI6IkhTMjU2In0.eyJleHAi
```

Figure 5.7. Access token string

The resource owner verifies the access token with the authorization server, and then authenticates the user as shown in figure 5. 8:



```
if (!id.IsAuthenticated)
    return
```

Figure 5. 8. After validating the access token, the user is authenticated

CHAPTER 6: CONCLUSION

We introduced an efficient and non-intrusive method to detect the face spoofing attacks utilizing only a single image. We applied a nonlinear diffusion based on an additive operator splitting (AOS) scheme to reveal the edges and surface texture in the input image. Previous approaches used handcrafted features, such as the LBP algorithm, to extract the information features from the input image. In contrast, this work proposes a specialized deep convolution neural network that can extract the complex features of the input diffused image to differentiate between a fake and real face. Our CNN has proven to be powerful in extracting not only the edges, but also the textures of the faces. We have achieved the highest reported accuracy of 99% on the widely used NUAA dataset. Our analysis of the misclassified faces indicated that over-exposures, blurring, and reflections affected our proposed approach's ability to detect spoofing attacks. In addition, we tested our method on the Replay Attack dataset utilizing only one frame of a replay video attack and achieved a HTER of 10% which is a better result when compared to previous static algorithms results a 10% HTER. Moreover, we developed a biometric authentication system based on web services to enhance OAuth authentication security. We built a Secure Login Service to authenticate the identity of the user that consists of four layers: an image registration service, a face liveness detection service, a verification service, and an access token service. These services are developed in the authorization server to verify the user and issue the access token. The entire web services are built based on RESTful architecture style. If the user is verified and authenticated, the access token is issued with a specific scope of authorization level.

REFERENCES

[1] F. B. Shaikh and S. Haider, "Security threats in cloud computing," in 2011 international conference for Internet technology and secured transactions (ICITST), 2011, pp. 214-219.

[2] P. Nadanam and R. Rajmohan, "QoS evaluation for web services in cloud computing," in 2012 Third International Conference on Computing Communication & Networking Technologies (ICCCNT), 2012, pp. 1-8.

[3] W. Meng-Yu and L. Tsern-Huei, "Design and implementation of cloud API access control based on OAuth," in 2013 IEEE TENCON Spring Conference, 2013, pp. 485-489.

[4] J. Wayman, A. Jain, D. Maltoni, and D. Maio, "An Introduction to Biometric Authentication Systems," in Biometric Systems: Technology, Design and Performance Evaluation, J. Wayman, A. Jain, D. Maltoni, and D. Maio, Eds., ed London: Springer London, 2005, pp. 1-20.

[5] L. Weiwen, "Face liveness detection using analysis of Fourier spectra based on hair," in 2014 International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR), 2014, pp. 75-80.

[6] J. Galbally, S. Marcel, and J. Fierrez, "Biometric Antispoofing Methods: A Survey in Face Recognition," IEEE Access, vol. 2, 2014, pp. 1530-1552.

[7] D. A. Socolinsky and A. Selinger, "A comparative analysis of face recognition performance with visible and thermal infrared imagery," in 16th International Conference on Pattern Recognition, 2002, pp. 217-222 vol.4.

[8] M. Pietikäinen and A. Hadid, "Texture features in facial image analysis," Advances in Biometric Person Authentication, ed: Springer, 2005, pp. 1-8.

[9] T. Choudhury, B. Clarkson, T. Jebara, and A. Pentland, "Multimodal person recognition using unconstrained audio and video," in International Conference on Audio-and Video-Based Person Authentication Proceedings, 1999, pp. 176-181.

[10] J. Maatta, A. Hadid, and M. Pietikainen, "Face spoofing detection from single images using micro-texture analysis," in 2011 International Joint Conference on Biometrics (IJCB), 2011, pp. 1-7.

[11] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, 2002, pp. 971-987.

[12] T. Ahonen, A. Hadid, and M. Pietikainen, "Face Description with Local Binary Patterns: Application to Face Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, 2006, pp. 2037-2041.

[13] I. Chingovska, A. Anjos, and S. Marcel, "On the effectiveness of local binary patterns in face anti-spoofing," in 2012 BIOSIG - Proceedings of the International Conference of the biometrics Special Interest Group (BIOSIG), 2012, pp. 1-7.

[14] K. Wonjun, S. Sungjoo, and H. Jae-Joon, "Face Liveness Detection From a Single Image via Diffusion Speed Model," IEEE Transactions on Image Processing, vol. 24, 2015, pp. 2456-2465.

[15] J. Weickert, B. M. T. H. Romeny, and M. A. Viergever, "Efficient and reliable schemes for nonlinear diffusion filtering," IEEE Transactions on Image Processing, vol. 7, 1998, pp. 398-410.

[16] Y. Jianwei, L. Zhen, L. Shengcai, and S. Z. Li, "Face liveness detection with component dependent descriptor," in 2013 International Conference on Biometrics (ICB), 2013, pp. 1-6.

[17] T. Xiaoyang and B. Triggs, "Enhanced Local Texture Feature Sets for Face Recognition Under Difficult Lighting Conditions," IEEE Transactions on Image Processing, vol. 19, 2010, pp. 1635-1650.

[18] X. Tan, Y. Li, J. Liu, and L. Jiang, "Face liveness detection from a single image with sparse low rank bilinear discriminative model," Computer Vision–ECCV 2010, ed: Springer, 2010, pp. 504-517.

[19] B. Peixoto, C. Michelassi, and A. Rocha, "Face liveness detection under bad illumination conditions," in 2011 18th IEEE International Conference on Image Processing (ICIP), 2011, pp. 3557-3560.

[20] K. Zuiderveld, "Contrast limited adaptive histogram equalization," in Graphics gems IV, 1994, pp. 474-485.

[21] Z. Zhiwei, Y. Junjie, L. Sifei, L. Zhen, Y. Dong, and S. Z. Li, "A face antispoofing database with diverse attacks," in 2012 5th IAPR International Conference on Biometrics (ICB), 2012, pp. 26-31.

[22] J. Li, Y. Wang, T. Tan, and A. K. Jain, "Live face detection based on the analysis of fourier spectra," Defense and Security, 2004, pp. 296-303.

[23] T. de Freitas Pereira, A. Anjos, J. M. De Martino, and S. Marcel, "LBP − TOP Based Countermeasure against Face Spoofing Attacks," in Computer Vision - ACCV 2012 Workshops: ACCV 2012 International Workshops, Daejeon, Korea, November 5-6, 2012, Revised Selected Papers, Part I, J.-I. Park and J. Kim, Eds., ed Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 121-132.

[24] Z. Guoying and M. Pietikainen, "Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, 2007, pp. 915-928.

[25] S. Bharadwaj, T. I. Dhamecha, M. Vatsa, and R. Singh, "Computationally Efficient Face Spoofing Detection with Motion Magnification," in 2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2013, pp. 105-110.

[26] P. Gang, S. Lin, W. Zhaohui, and L. Shihong, "Eyeblink-based Anti-Spoofing in Face Recognition from a Generic Webcamera" in IEEE 11th International Conference on Computer Vision, 2007, pp. 1-8.

[27] W. Di, H. Hu, and A. K. Jain, "Face Spoof Detection with Image Distortion Analysis," IEEE Transactions on Information Forensics and Security, vol. 10, 2015, pp. 746-761.

[28] A. K. Singh, P. Joshi, and G. C. Nandi, "Face recognition with liveness detection using eye and mouth movement," in 2014 International Conference on Signal Propagation and Computer Technology (ICSPCT), 2014, pp. 592-597.

[29] K. Sooyeon, Y. Sunjin, K. Kwangtaek, B. Yuseok, and L. Sangyoun, "Face liveness detection using variable focusing," in 2013 International Conference on Biometrics (ICB), 2013, pp. 1-6.

[30] K. Younghwan, Y. Jang-Hee, and C. Kyoungho, "A motion and similarity-based fake detection method for biometric face recognition systems," IEEE Transactions on Consumer Electronics, vol. 57, 2011, pp. 756-762.

[31] S. Tirunagari, N. Poh, D. Windridge, A. Iorliam, N. Suki, and A. T. S. Ho, "Detection of Face Spoofing Using Visual Dynamics," IEEE Transactions on Information Forensics and Security, vol. 10, 2015, pp. 762-777.

[32] Y. Junjie, Z. Zhiwei, L. Zhen, Y. Dong, and S. Z. Li, "Face liveness detection by exploring multiple scenic clues," in 2012 12th International Conference on Control Automation Robotics & Vision (ICARCV), 2012, pp. 188-193.

[33] B. Wei, L. Hong, L. Nan, and J. Wei, "A liveness detection method for face recognition based on optical flow field," in IASP International Conference on Image Analysis and Signal Processing, 2009, pp. 233-236.

[34] A. Anjos, M. M. Chakka, and S. Marcel, "Motion-based counter-measures to photo attacks in face recognition," Biometrics, IET, vol. 3, 2014, pp. 147-158.

[35] K. Kollreider, H. Fronthaler, and J. Bigun, "Non-intrusive liveness detection by face images," Image and Vision Computing, Elsevier, vol. 27, 2009, pp. 233-244.

[36] N. Erdogmus and S. Marcel, "Spoofing in 2D face recognition with 3D masks and anti-spoofing with Kinect," in 2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS), 2013, pp. 1-6.

[37] N. Erdogmus and S. Marcel, "Spoofing Face Recognition With 3D Masks," IEEE Transactions on Information Forensics and Security, vol. 9, 2014, pp. 1084-1097.

[38] N. Kose and J. L. Dugelay, "Countermeasure for the protection of face recognition systems against mask attacks," in 2013 10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG), 2013, pp. 1-6.

[39] R. Wallace, M. McLaren, C. McCool, and S. Marcel, "Inter-session variability modelling and joint factor analysis for face authentication," in 2011 International Joint Conference on Biometrics (IJCB), 2011, pp. 1-8.

[40] D. Hardt. (2013) "The OAuth 2.0 Authorization Framework (RFC6749)," http://tools.ietf.org/html/rfc6749.

[41] B. Lakshmiraghavan, Pro ASP.NET Web API Security: Securing ASP.NET Web API. in Apress, 2013.

[42] Y. Feng and S. Manoharan, "A security analysis of the OAuth protocol," in 2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM), 2013, pp. 271-276.

[43] A. P. Leinonen, T. Tuikka, and E. Siira, "Implementing Open Authentication for Web Services with a Secure Memory Card," in 2012 4th International Workshop on Near Field Communication (NFC), 2012, pp. 31-35.

[44] I. A. Gomaa, G. I. Salama, and I. F. Imam, "Biometric OAuth service based on finger-knuckles," in 2012 Seventh International Conference on Computer Engineering & Systems (ICCES), 2012, pp. 170-175.

[45] A. K. Jain, A. Ross, and S. Pankanti, "Biometrics: a tool for information security," IEEE Transactions on Information Forensics and Security, vol. 1, 2006, pp. 125-143.

[46] A. N. Kataria, D. M. Adhyaru, A. K. Sharma, and T. H. Zaveri, "A survey of automated biometric authentication techniques," in 2013 Nirma University International Conference on Engineering (NUiCONE), 2013, pp. 1-6.

[47] A. Alotaibi and A. Mahmmod, "Enhancing OAuth services security by an authentication service with face recognition," in 2015 IEEE Long Island Systems, Applications and Technology Conference (LISAT), 2015, pp. 1-6.

[48] A. P. Witkin, "Scale-space filtering," ed: Google Patents, 1987.

[49] J. J. Koenderink, "The structure of images," Biological cybernetics, vol. 50,1984, pp. 363-370.

[50] J. Canny, "A Computational Approach to Edge Detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-8, 1986, pp. 679-698.

[51] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, 1990, pp. 629-639.

[52] J. Weickert, B. T. H. Romeny, and M. Viergever, "Efficient and reliable schemes for nonlinear diffusion filtering," IEEE Transactions on Image Processing, vol. 7, 1998, pp. 398-410.

[53] J. Ralli. (2014) "PDE Based Image Diffusion and AOS," http://www.jarnoralli.fi/joomla/images/pdf/non_linear_image_diffusion_and_aos_ralli_2014.pdf.

[54] E. H. Land and J. McCann, "Lightness and retinex theory," JOSA, vol. 61, 1971, pp. 1-11.

[55] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, 1998, pp. 2278-2324.

[56] B. B. Le Cun, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," in Advances in neural information processing systems, 1990.

[57] C. Garcia and M. Delakis, "Convolutional face finder: a neural architecture for fast and robust face detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, 2004, pp. 1408-1423.

[58] S. Lawrence, C. L. Giles, T. Ah Chung, and A. D. Back, "Face recognition: a convolutional neural-network approach," IEEE Transactions on Neural Networks, vol. 8, 1997, pp. 98-113.

[59] B. Fasel, "Robust face analysis using convolutional neural networks," in 16th International Conference on Pattern Recognition, 2002, pp. 40-43.

[60] A. Alotaibi and A. Mahmood, "Deep face liveness detection based on nonlinear diffusion using convolution neural network," Signal, Image and Video Processing, Springer, 2016, pp. 1-8.

[61] R. Goroshin and Y. LeCun, "Saturating auto-encoders," arXiv preprint arXiv:1301.3577, 2013.

[62] S. Bengio and J. Mariéthoz, "A statistical significance test for person authentication," in ODYSSEY04-The Speaker and Language Recognition Workshop, 2004.

[63] A. Alotaibi and A. Mahmood, "Enhancing computer vision to detect face spoofing attack utilizing a single frame from a replay video attack using deep learning," in 2016

International Conference on Optoelectronics and Image Processing (ICOIP), 2016, pp. 1-5.

[64] M. Shehab and S. Marouf, "Recommendation Models for Open Authorization," IEEE Transactions on Dependable and Secure Computing, vol. 9, 2012, pp. 583-596.

[65] L. Ke and X. Ke, "OAuth Based Authentication and Authorization in Open Telco API," in 2012 International Conference on Computer Science and Electronics Engineering (ICCSEE), 2012, pp. 176-179.

[66] J. Wayman, A. Jain, D. Maltoni, and D. Maio, "An introduction to biometric authentication systems," in Biometric Systems, ed: Springer, 2005, pp. 1-20.

[67] H. Di, S. Caifeng, M. Ardabilian, W. Yunhong, and C. Liming, "Local Binary Patterns and Its Application to Facial Image Analysis: A Survey," IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, vol. 41, 2011, pp. 765-781.