Republic of Iraq

Ministry of Higher Education and Scientific Research

University of Babylon

College of Information Technology

Department of Software

Object Behavior Analysis-Based Deepfake Video Prediction Using Deep Learning

A Dissertation

Submitted to the Council of the College of Information Technology, University of Babylon in Partial Fulfillment of the Requirements for the Doctor of Philosophy Degree in Information Technology/ Software

By

Qasim Jaleel Khudhair Abyss

Supervised by

Prof. Dr. Israa Hadi Ali

2023 A.D. **1445 A.H.**

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

۞ يُخَادِعُونَ اللَّهَ وَالَّذِينَ آمَنُوا
وَمَا يَخْدَعُونَ إِلَّا أَنفُسَهُمْ وَمَا
يَشْعُرُونَ ۞

صدق الله العلي العظيم

سورة المجادلة (الآية ١١)

Supervisor Certification

I certify that this dissertation entitled **" Object Behavior Analysis-Based Deepfake Video Prediction Using Deep Learning"** was prepared under my supervision at the Department of Software / College of Information Technology / University of Babylon, by **Qasim Jaleel Khudhair Abyss** as a partial fulfillment of the requirements of the degree of **Ph.D. in Information Technology / Software**.

Signature:

Name: **Prof. Dr. Israa Hadi Ali**

Title: **Professor**

Date: / / 2023

The Head of the Department Certification

In the view of available recommendations, I forward the dissertation entitled **"" Object Behavior Analysis-Based Deepfake Video Prediction Using Deep Learning "** for debate by the examination committee.

Signature:

Name: **Dr. Sura Zaki Alrashid**

Title: **Asst. Prof.**

Date: / /2023

Declaration

I hereby declare that this Dissertation, submitted to University of Babylon in partial fulfillment of requirement for the degree of Ph.D. in Information Technology \ Software, has not been submitted as an exercise for a similar degree at any other University. I also certify that this work described here is entirely my own except for experts and summaries whose source are appropriately cited in the references.

Signature:

Name: Qasim Jaleel Khudhair

Date: /   /2023

DEDICATION

I dedicate this effort

TO

Dears

My father,

My mother,

With all my gratitude

Qasim

Acknowledgements

All the praises and thanks are to Allah, the most beneficent, the most merciful, for his graces that enabled me to continue the requirements of my study.

I would like to express my deepest appreciation to my supervisor ***Prof. Dr. Israa Hadi Ali*** for her invaluable guidance, supervision and untiring efforts during the course of this work.

Sincere appreciation is due to my family especially my dear mother and my father for their patience, encouragement and help during the work.

Finally I would like to thank all the kind, helpful and lovely people who helped me directly or indirectly to complete this work and I apologize for not being able to mention them all the names, but their favour will always be cherished.

Qasim✍

Abstract

Deep fake is a term used to describe the production and manipulation of manufactured or fake movies, sounds, or images using artificial intelligence (AI) methods. It entails using massive datasets of real media to learn AI models, particularly generative adversarial networks (GANs). These models can then produce fresh, synthetic content that mimics real content but is fake. There are types of deep fake, including the transmission of facial expressions, in which one person's facial expressions are projected onto another person's face in a video. It enables altering a person's emotional expressions, causing them to exhibit feelings other than those recorded. To impersonate someone, identity thieves create movies that make that person look different. The problem with the traditional deep fake detection methods depends on the artifacts found in fake videos. When there are videos that do not contain artifacts, they were created very close to real videos, as the traditional methods do not give good results. The aim of the dissertation, the detection of deep fakes, is based on detecting the object's behavior in terms of facial expressions in the videos created in an ideal way for fakes. Each person has special behaviors when speaking and facial expressions such as sadness, anger, and others. This feature can be exploited to detect deep fakes by comparing people's behavior using facial expressions such as facial action units and face poses.

The proposed system consists of two main phases. The first phase is to extract features from facial expressions and facial poses. We take frames as input from real and fake videos and extracts their features. The most important features extracted from facial expressions are facial poses and action units. Each action unit is a specific movement of the facial muscles, which is one of the important features in distinguishing the object's behavior. The pretrained JAA-Net model extracts the action unit, as the number of action units it extracts is twelve. The pretrained Hybrid Face Pose model extracts

the three features of the facial pose, namely yaw, pitch, and roll. The second phase predicts real and fake videos using the features collected from the first phase and then entered into the prediction model. The prediction model used is BiLSTM, which is trained with the features. After training, the model can predict whether the videos are real or fake.

The datasets used in the proposed system are the Barack Obama dataset and Forensics++. Barack Obama's dataset contains videos considered effective and very close to reality. Barack Obama's dataset was relied upon in the prediction model training process. The forensics++ dataset was also used for training and comparing the obtained results with the results of the Barack Obama dataset. The proposed system proved to be highly accurate compared to traditional methods, and the accuracy of the proposed system reached 99.403%.

Index

Table of Contents

CHAPTER ONE INTRODUCTION

CHAPTER TWO THEORETICAL BACKGROUND

CHAPTER THREE THE PROPOSED SYSTEM

CHAPTER FOUR RESULTS AND DISCUSSION

List of Tables

List of Figures

List of Algorithms

List of Abbreviations

Abbreviation	Meaning
AU	Action Unit
AUC	Area Under Curve
biGAN	Bidirectional Generative Adversarial Networks
Bi-LSTM	Bidirectional Long Short-Term Memory
CM	Confusion Matrix
CNN	Convolutional Neural Network
DARPA	Defense Advanced Research Projects Agency
DBN	Deep Belief Network
DL	Deep Learning
EMFACS	Emotion Facial Action Coding System
FACS	Facial Action Coding System
FPN	Feature Pyramid Network
GANs	Generative Adversarial Networks
GRU	Gated Recurrent Unit
k-NN	K-Nearest Neighbor
log Loss	Logarithmic Loss
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MSE	Mean Squared Error

RBF	Radial Basis Function
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
RPN	Region Proposal Network
SMOTE	Synthetic Minority Oversampling Technique
SVM	Support Vector Machine
YOLO-CRNN	You Only Look Once Convolution Recurrent Neural Network

Chapter One

GENERAL INTRODUCTION

General Introduction

1.1 Introduction

Fake videos with facial information created through digital modification, particularly using deep fake technologies, have recently become a major public concern [1]. The phrase "Deepfake" refers to a deep learning-based technology for creating fake videos by exchanging a person's face with another person's face. In addition, fake content includes fake news, hoaxes, and financial fraud [2]. Since anyone can now use the technology, a lot of fake videos have been shared on social media. Deepfake is a term for digital media that has been changed, like an image or video of a person that has been changed to look like someone else. It is a problem that is getting worse in today's world[3]. It has often been used to put the faces of well-known Hollywood stars in pornographic videos. It is also used to give politicians false information and rumors. In 2018, a fake video of Barack Obama is created to put words he never uttered [4]. In addition, in the US 2020 election, deepfakes have already been used to manipulate Joe Biden's videos showing his tongue out[4][5]. Deepfakes encompass various manipulation techniques for facial features in images and videos[6][7]. These include face swap deep fakes, which replace one person's face with another's; lip-sync deep fakes that modify lip movements to sync with different audio; expression deep fakes that alter emotions displayed on a person's face; gender-swap deep fakes that transform a person's gender appearance; age progression/regression deep fakes for simulating aging; makeup deep fakes for altering appearance with cosmetics; face morphing deep fakes blending features of two faces; superimposition deep fakes for seamless overlaying; and even tutorials simulating makeup transformations or celebrity impersonation deepfakes[8].

Generative adversarial networks (GANs) are generative and sophisticated deep learning technologies that can be applied to generate fake images and videos that are hard for a human to identify from the true ones[9][10]. These models are trained on a set of data, and then they are used to make fake pictures and videos. This kind of deepfake model requires a large set of training data for those deepfakes[11][12]. The more data the model has, the more real and convincing the images and videos it can make[13][14]. The fact that there are a lot of videos of presidents and Hollywood stars on social media can help people make fake news and rumors sound real and can have serious effects on our society[15][16]. Recent studies show that deepfake images and videos are being shared a lot on social media[12]. So, finding deepfake videos and images has become more and more crucial. To get more people interested in research, groups like the US Defense Advanced Research Projects Agency (DARPA), Facebook Inc. and Google launched a research initiative to detect and prevent deepfakes [17][18]. As a result, many deep learning approaches such as long short-term memory (LSTM), recurrent neural network (RNN), and even the hybrid approaches have been proposed in order to detect deep fake images and videos and to bring up more research on this field [19][20].

The deep learning algorithms used in deep fake detection are feature-based, and one such feature that is used in deep fake detection is facial expressions. Facial expressions are indeed crucial in human communication and play a significant role in conveying emotions, intentions, and understanding during interactions. It is important to figure out if the expressions on a person's face in an image or video are natural or if they have been changed by someone else[21][22]. The moving parts of the face are the most important parts of a facial expression. Action units (AUs) are the basic things that certain muscles or groups of muscles do. In the Emotion Facial Action Coding System (EMFACS), Ekman and Friesen looked into the relationship between AU movement and facial emotions[23][24]. They found that all AUs are external representations of muscle movements. Face pose analysis is the process of identifying the location and orientation of a face (Yaw, Pitch, and Roll) and facial analysis[25][26].

Deepfake detection methods are generally divided into two categories: artifact-specific and undirected approaches [27]. Artifact-specific deepfakes frequently produce artifacts that are difficult for humans to detect but can be discovered using machine learning and forensic analysis [28] [29]. Some tools look for certain artifacts to spot fakes. There are several kinds of artifacts: temporal artifacts in behavior, physiology, synchronization, and coherence; spatial artifacts in blending, environments, and forensics [6]. And the second category is undirected approaches, where some researchers train deep neural networks as general classifiers and let the network choose which attributes to examine, as opposed to concentrating on a particular artifact [30]. The two main methodologies used by researchers are classification and anomaly detection.

1.2 Problem statement

The deepfake has become an important discussion topic in society because of the permanent dangers this fake video poses to individuals, communities, and countries. Fake videos created by deep learning are very close to reality. Deepfake videos are perfect and difficult to detect with traditional methods that rely on clear visual effects. These videos do not contain artifacts and cannot be distinguished as real or fake because they are created perfectly. The traditional methods relied mainly on the artifacts in the deep fake videos, such as blending, environments, and forensics. One of the most common deep learning networks that create deep fakes is GAN. The GAN network produces deep fake videos that look more realistic. Moreover, other types of GAN have been developed, such as cycle GAN, biGAN, couple GAN, etc.

Another problem in deep fake detection is the diversity of methods that creates deep fakes, in addition to the great and continuous development of the tools used in the process of creating fakes in different parts of the face. This diversity of methods makes it difficult to use traditional methods in the detection of deep fakes, as well as not giving good and accurate results.

1.3 Literature Review

Some of the works that are related to the proposed work are reviewed below:

- In November 2018, Xin Yang et al. [31] used the hypothesis that deep fakes were produced by splicing a synthetic face region into the original image, creating faults in the process that may be seen when 3D head postures were calculated from the face photos. The 3D head posture reflects how the world coordinates were rotated and translated to the matching camera coordinate. A set of actual face photos and deep fakes were used to evaluate an SVM classifier's performance utilizing features based on this cue. The measure used in the research was the accuracy as a performance indicator, and its value was 0.89%.
- In September 2018, Darius Afchar et al. [32] offered a strategy to quickly and effectively spot face tampering in films and focused, in particular, on Deepfake and Face2Face datasets. Due to the compression, which severely degrades the data, traditional image forensics techniques were typically not well suited to videos. Concentrating on the mesoscopic characteristics of images, they used a deep learning approach and offered two networks (Meso-

4 and MesoInception-4), each with a small number of layers. The detection rate was 98% for Deepfake and 95% for the Face2Face dataset.

- In 2019, Shruti Agarwal and Hany Farid [33] found that it is possible to model facial gestures and expressions that characterize a person's speech pattern. The model postulates that a person's facial expressions and body language change as they speak in distinctive (though probably not unique) ways. It begins by tracking facial and head movements from a single video input before isolating the presence and potency of particular action units. Then, it creates a novelty detection model (one-class support vector machine, or SVM) that can tell a real person from a comedy impersonator and a deep-fake impersonator. The dataset used was Barack Obama's, and the accuracy of the method was 94%.
- In December 2020, Shruti Agarwal et al. [34] described a biometric-based forensic method for identifying face-swap deep fakes. In this method, a temporal behavioral biometric based on head movements and facial expressions was combined with a static biometric based on facial recognition. The behavioral embedding was trained using a CNN with a metric learning objective function. It demonstrates the effectiveness of this method using various sizable video datasets and real-world deep fakes. The datasets used in research were WLDR, Face Forensics++, and DFD, and the accuracy ranges from 0.93 % to 0.99%.

- In January 2020, Ruben Tolosana et al. [35] offered a thorough evaluation of deep fake generations in terms of facial regions and the effectiveness of fake detection. The segmentation of the full face as input to the fake detection system and the segmentation of only particular facial regions were two separate methodologies that were examined. For the second strategy, four distinct facial regions were chosen: the eyes, nose, mouth, and rest (i.e., the part of the face obtained after removing the eyes, nose, and mouth from the entire face). The Xception network was employed in the study, and it offers excellent results for false detection. The datasets used in this research were UADFV, Celeb-DF, Forensics++, and DFDC. The accuracy ranges from 82.46% to 99.40%.

- In October 2021, Sreeraj Ramachandran et al. [29] used various loss functions and deep fake-generating approaches to fully assess the effectiveness of deep facial recognition in detecting deep fakes. Deep face recognition was more effective at identifying deepfakes than two-class CNNs and the ocular modality, according to experimental studies on the Celeb-DF and Face Forensics++ deepfake datasets. According to reported results, using face recognition on the Celeb-DF dataset made it possible to detect deep fakes with an area under the curve (AUC) of up to 0.98%.
- In April 2021, Robail Yasrab [19] proposed a method for automatic deep fake media detection using the LSTM network in body language analysis. In this study, it was proposed that upper body language analysis be used as a deep fake detection tool. In particular, a classification model for deep fake detection was created and trained using a many-to-one LSTM network. The hyperparameters were changed while training various models to create a final model with benchmark accuracy. The accuracy of the method was 94.39%.

- In April 2021, Thomas Reolon [36] used the face detection method OpenFace to extract important details from movies of a specific individual, such as facial landmarks and head poses. After that, a pipeline of 15 one-class classifiers will be trained on real footage of a particular individual before being tested on real and fake ones. The dataset used in this study was Barack Obama's, and the method's accuracy was 0.896%.

- In September 2022, Aya Ismail et al. [37] presented You Only Look Once Convolution Recurrent Neural Networks (YOLO-CRNNs), a study to identify deepfake films. A tuned EfficientNet-B5 was used to extract the spatial attributes of these faces after the YOLO-Face detector had detected face regions in each frame of the movie. These features were supplied as a batch of input sequences into a Bidirectional Long Short-Term Memory (Bi-LSTM) to extract the temporal features. The novel method was tested on CelebDF-faceForencics++ (c23), a new, sizable dataset combining Celeb-DF and faceForencics++ (c23), two widely used datasets. The accuracy of the research was 0.8938%.
- In December 2022, Matyas Bohcek and Hany Farid [38] employed an identity-based methodology in the study to protect global leaders from deep fake imposters who practice that on numerous real-world video residences. This method records distinctive facial (gestural, head stance, eyes stare), gestural, and vocal habits that can tell a world leader from a profound fake imposter or imitation by deep learning. The datasets used in this research were World Leaders and Forensics++. The accuracy ranges were from 82.80 to 0.9521.
- In August 2022, Christeen T. Jose's [39] findings on using temporal pictures for deep fake detection were presented in this proposal by creating an image (referred to as a temporal image) utilizing the pixel values at these face landmarks. It was able to represent the temporal relations that arise in the movement of 468 facial markers over frames of a given film as spatial relations. The spatial correlations between the pixels in an image can be recognized using CNNs. For the investigation, 10 distinct ImageNet models were taken into account. The dataset used in this paper was Forensics++, and the method's accuracy was 0.94834%.

Most of the research mentioned in the related works cannot deal with the deep fake videos that are created perfectly. Most of the research depends on the artifacts found in the videos. This dissertation relies on the behavior of the object, which depends on facial expressions that differ from one person to another and according to emotional states. It is important to figure out if the expressions on a person's face in an image or video are natural or if they have been changed by someone else. Table 1.1 illustrates a summary of related works of deepfake detection.

Table 1.1. Summary of the Related Works of Deepfake Detection

Reference	Prediction method	Features Type	Dataset	Results
[31]	SVM classifier	3D head poses	UADFV	ROC= 0.89
[32]	Meso-4 ,and MesoInception-4 networks	Mesoscopic properties	Face2Face dataset, Deepfake dataset	detection rate = 98% for Deepfake 95% for Face2Face
[33]	one-class SVM	Facial action unit and head movement	Barack Obama	Acc. = 0.94
[34]	CNN(Facial Attributes-Net, FAb-Net)	Appearance and Behavior	WLDR, Face Forensics++,DFD	Acc. = From 0.93 To 0.99
[35]	Pre-training Xception network	facial regions	UADFV, Celeb-DF, Forensics++, DFDC	Acc. = From 82.46 To 99.40

[29]	Pre-training ResNet-50	Artifact of facial	Celeb-DF, Forensics++	Acc. = 0. 99
[19]	LSTM	upper body language analysis	Barack Obama	Acc. = 0.9439
[36]	SVM	Facial Landmark and head pose	Barack Obama	Acc. = 0.896
[37]	YOLO-CRNNs, Bi-LSTM	Face regions	Celeb-DF, Forensics++	Acc. = 0.8938
[38]	SVM	Artifact of facial (gestural, head stance, eyes)	World leaders, Forensics++	Acc. = From 82.80 To 0.9521
[39]	Pre-training InceptionResNetV2	landmarks	Forensics++	Acc. = 0.94834

1.4 Challenges of Dissertation

Many research efforts have been made to identify fake videos on the internet or social media. However, developing a deep fake video detection model has proven to be a challenging task for several reasons, including the evolution of deep fake video creation technologies where there is little in the data set that contains almost perfect (real and fake) videos. Deep fake cannot be located in perfectly created videos that do not contain artifacts. Also, the deep fake of the videos is very realistic, closer to the truth, and characterized by very high accuracy.

On the other hand, the diversity of algorithms that generate facial manipulation means that each algorithm generates deep fakes in a different way. Also, these algorithms depend on an architecture whose function is to change either the entire facial expression or part of it. This makes it difficult to determine where the manipulation is. The publicly available datasets are not appropriate for this research as they do not fit the proposed face and landmark requirements. Most of the datasets contain short and limited videos of people. These datasets make it difficult to extract all kinds of action units and head poses. It also causes inaccuracies when the network trains deep learning to identify real and fake videos.

In addition, the method needs a dataset that contains different movements of the object in order to train the network on the different behaviors of the object, which helps us discriminate between real videos and fakes.

1.5 Dissertation Objectives

The main aim of this dissertation is to build a deep learning model that can perfectly detect well-created deep fakes that traditional methods cannot detect with high accuracy. In order to achieve this aim, many objectives are marked:

- Building a developed model for deep fake detection based on human behavior, which varies from person to person in terms of facial expressions. Facial expressions differ in cases of anger, happiness, sadness, and others.
- Extraction of facial expression features from the person's action unit and head pose using deep learning.
- Proposing the architecture of deep neural networks to improve prediction accuracy through analyzing behavior and facial expressions based on a dataset.
- Another objective is to increase the features extracted from the fake dataset by converting the data from imbalanced to balanced data, which increase the prediction accuracy of the proposed system.

1.6 Dissertation Contributions

This dissertation has achieved the following contributions in comparison to previous work:

- Building a model based on facial expressions to detect deep fake, and these facial expressions are similar to a biometric. This method represents a good and accurate method for detecting deep fakes created in an ideal way.
- Adapting the pre-trained JAA-Net model to the proposed deep learning model by extracting twelve action units from facial expressions. Adaptation of the hybrid face pose model by adding the pre-trained Resnet18 model, through which three features are extracted from the face pose. These features are extracted from the face pose, and the facial action unit increased the accuracy of the proposed model.
- Build and implement the BiLSTM deep neural network architecture to adapt to the requirements of the proposed system. The adaptation is done by adding batch normalization layers and an L1 regularization factor, which increases the prediction accuracy of real and fake videos.

1.7 Dissertation Organization

In addition to Chapter One, the research contains the following chapters for different purposes:

Chapter Two, *"Theoretical Background"*: Many concepts will be introduced in this chapter, including datasets used in this dissertation, Facial Expression, Face Pose, Deep learning, transfer learning, and evaluation measures.

Chapter Three, ***"Methodology"***: This chapter first views the proposed methodology. Then, two parts will be discussed. The first part presents feature extraction. Furthermore, the second part explains the prediction using bidirectional LSTM.

Chapter Four, "Results and Discussion": displays the results of the proposed models on the datasets used in this dissertation. Then, the results will be compared with the previous studies.

Chapter Five, ***"Conclusions and Future Works"***: Many conclusions drawn from our proposed models and suggestions for future work will be introduced.

Chapter Two

THEORETICAL BACKGROUND

CHAPTER TWO
THEORETICAL BACKGROUND

2.1 Overview

This chapter introduces the definition of facial expressions, the facial action unit, and the types of methods for detecting the facial action units. It also provides definitions of facial landmarks and ways to detect them. The chapter then deals with the definitions of face alignment and face pose and how to represent them. It also covers the networks used in deep learning, transfer learning, which will later be used in landmark detection, the action unit, and predictions. Next, it presents classification using machine learning and its types. Finally, it shows the methods used in the performance measures.

2.2 Datasets and Types of Data Augmentation

The initial phase or foundational step in any machine learning or deep learning endeavor involves the acquisition of data. A dataset is a set of samples used to train, validate, and test machine learning models in deep learning. In machine learning, a dataset typically consists of two main components: the input data (features), which represent the information or attributes of the examples, and the corresponding goal labels (ground truth), which specify the correct or desired output for each example. Datasets are essential for training precise and efficient deep learning models because they give the model the knowledge it needs to recognize patterns and correlations in the data [40].

This dissertation focuses on a deep fake dataset of videos. A deep learning dataset is a curated set of data used to train and test deep learning models that are intended to produce or recognize fake content. Deepfakes are modified materials, such as photos, videos, or audio samples, that make use of artificial intelligence, especially deep learning algorithms, to produce content that seems authentic but is actually manufactured[41]. Deepfake datasets are used for a variety of tasks, such as creating algorithms to identify deepfakes and advancing research in multimedia forensics[42][43]. They are also used to train models that produce deep fakes. These datasets are essential for examining the difficulties and vulnerabilities involved in the manipulation of media material as well as for enhancing the precision of deep fake generation and detection models[44].

Researchers and developers can use different deepfake datasets that are available to the public for different reasons. Some of these types of datasets are:

a. DeepFake Detection Dataset (DFDD)

The DFDD is a set of videos put together by Facebook AI. They have both real and "deep fake" material[16].

b. Celeb-DF Dataset

The Celeb-DF collection has pictures of famous people's faces that are very clear. Even though it was not made for deepfakes in particular, it has been used as a source of high-quality pictures of faces for making deepfakes[45].

c. Forensics++ Dataset

Forensics++ is a set of videos that were changed and made using different deepfake methods. It talks about many different ways to change a person's face, which makes it useful for study into finding deepfakes[46].

d. Barack Obama videos Dataset

These videos are collected with high pixel quality. Video files of two types, original and synthetic, were collected online to create a dataset for Barack Obama. Original video clips were downloaded from the official Miller Center website. This site contains US presidential speeches. The video formats are MP4 with 30 fps quality and different video sizes. Deep fake videos of Barack Obama collected from YouTube[33].

Data augmentation is a technique utilized frequently in machine learning, particularly in the disciplines of computer vision and natural language processing, to artificially increase the diversity and size of a dataset by transforming the original data samples. Data augmentation aims to enhance the generalization and robustness of machine learning models by exposing them to a wider variety of input data variations[44].

Data augmentation in computer vision frequently entails applying various transformations to images. The key concept is to generate new examples that are still meaningful and relevant to the problem at hand while introducing sufficient variation to help the model acquire a more comprehensive set of features and patterns[47][48]. Some common image augmentation techniques include:

- **Random Flipping**: Flip the image horizontally or vertically to create a mirror image[49].
- **Rotation**: Randomly rotate the image's angle of rotation[50].
- **Scaling**: Resize the image based on a specified factor.
- **Crop and Resize**: Crop a portion of the image before resizing it to its original proportions[51][52].
- **Color Adjustments**: Adjust the image's luminance, contrast, saturation, and hue[53].
- **Gaussian Noise**: Small quantities of Gaussian noise should be added to the image[54].
- **Random Erasing**: Randomly obliterate rectangles from the image[54].

- **The Synthetic Minority Oversampling Technique (SMOTE)**: SMOTE is a data augmentation technique used to address class imbalance in machine learning, specifically in binary classification problems[55]. It seeks to balance the class distribution by generating synthetic samples for the minority class, thereby decreasing the bias toward the majority class and enhancing the model's performance on the minority class[56]. In class-imbalanced datasets, where one class (the minority class) has substantially fewer instances than the other class (the majority class), machine learning models can develop a majority-class bias[57][58]. This can result in subpar performance for the minority class, which is frequently the class of interest in many applications. Mathematically, the process of generating a synthetic sample can be represented as follows:

$$\text{Synthetic Sample} = \text{Sample} + \text{Random\_Number} * (\text{Neighbor} - \text{Sample}) \ldots (2.1)$$

Where Sample represents an instance from the minority class, Random_Number is between 0 and 1, neighbor is other instances from the minority class.

2.3 Facial Expression

Facial expressions are the facial movements and alterations that communicate a variety of emotions, thoughts, and intentions. They are a vital component of human communication and play a crucial function in expressing information to others. According to the research of psychologist Paul Ekman, there are six widely recognized facial expressions: happiness, sadness, fear, wrath, surprise, and disgust, as shown in Figure (2.1) [24]. Certain expressions are associated with distinct facial muscle movements, such as the elevation of the commissers in response to happiness, the furrowing of the brow in response to grief, and the widening of the eyes in response to surprise [59] [60].

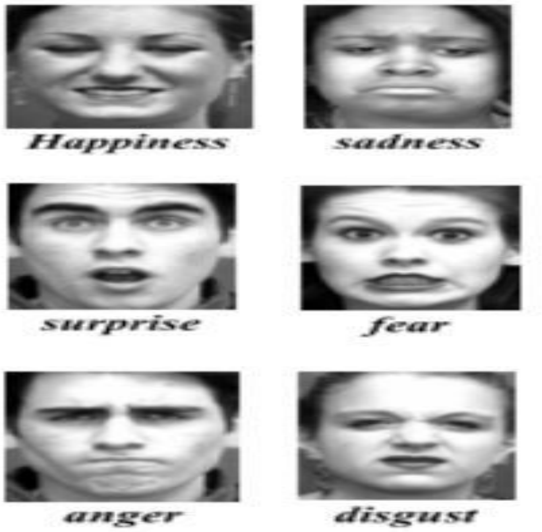


Figure (2.1): The Six Basic Expressions[60]

The facial expressions are not always deliberate and might be prompted by external stimuli or internal emotional states. For instance, when a person encounters a frightening object or event, their facial expression may reveal dread, even if they attempt to conceal it. Moreover, facial expressions can convey more sophisticated emotions and social indicators, such as curiosity, boredom, contempt, and empathy. These expressions may involve tiny changes in facial muscles, such as a little brow lift or lip tightness. Individual and cultural factors can influence facial expressions. Different cultures may have various rules surrounding the expression of emotions, and individuals may have varying degrees of expressiveness or facial control[61]. In addition to transmitting emotions and social cues, facial expressions can convey vital information about a person's physiological health. A warm face, for example, can signify embarrassment or wrath, but a pale face can imply fear or shock. They are essential components of human communication, serving as a nonverbal manner of transmitting emotions, thoughts, and intentions. Overall, they are an important part of human communication because they are a way to show emotions, thoughts, and intentions without using words [62].

The technique of recognizing and deciphering the emotions expressed by a person's facial expressions is known as facial expression detection. FER generic pipeline (in the case of conventional learning) as shown in Figure (2.2)[61]. Facial expressions can be recognized using a variety of techniques, including:

- Techniques for computer vision: The field of artificial intelligence known as computer vision focuses on giving computers the ability to analyze and understand visual data. The location of the eyes, mouth, and nose can be detected and extracted using computer vision techniques[63].
- Algorithms for machine learning: Algorithms for machine learning can classify face expressions into one or more emotional groups. To increase the accuracy of these systems, massive datasets of labeled facial expressions can be used as training data[64].
- Facial action coding system (FACS): Paul Ekman and Wallace Friesen created this method in the 1970s to categorize the distinct facial muscle movements that are connected to various emotions. The emotional state of the individual can be properly determined by FACS by examining these muscle movements[65] [66].

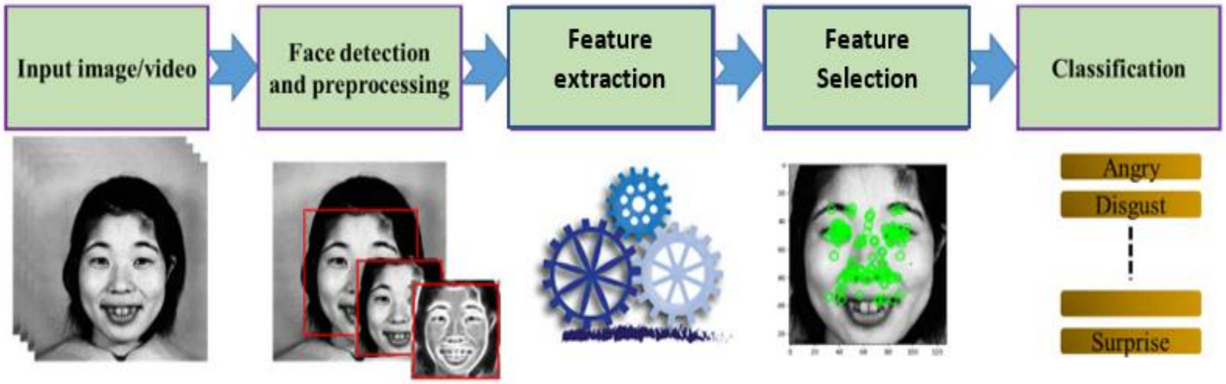


Figure (2.2): Facial Expression Detection Generic Pipeline

2.3.1 Facial Action Coding System (FACS)

A thorough, anatomically based approach for tracking all visually perceptible facial movement is the Facial Action Coding System (Ekman & Friesen, 1978; Ekman, Friesen & Hager, 2002)[67]. On the basis of 44 distinct action units (AUs), as well as various categories of head and eye postures and motions, FACS characterizes all visually discernible facial activity, as shown in Table (2.1). FACS defines a set of facial action units (AUs), each corresponding to a specific facial muscle or group of muscles and representing a distinct facial expression[65][68].

Table (2.1): Types Action Units in the Facial Action Coding System[69]

No.	AU number	Descriptor	No.	AU number	Descriptor
1	AU 1	Inner Brow Raiser	23	AU 24	Lip Pressor
2	AU 2	Outer Brow Raiser	24	AU 25	Lips Part
3	AU 4	Brow Lowerer	25	AU 26	Jaw Drop
4	AU 5	Upper Lid Raiser	26	AU 27	Mouth Stretch
5	AU 6	Cheek Raiser	27	AU 28	Lip Suck
6	AU 7	Lid Tightener	28	AU 29	Jaw Thrust
7	AU 8	Lips Toward Each Other	29	AU 30	Jaw Sideways
8	AU 9	Nose Wrinkler	30	AU 31	Jaw Clencher
9	AU 10	Upper Lip Raiser	31	AU 32	Lip Bite
10	AU 11	Nasolabial Fold Deepener	32	AU 33	Blow
11	AU 12	Lip Corner Puller	33	AU 34	Puff
12	AU 13	Cheek Puffer	34	AU 35	Cheek Suck
13	AU 14	Dimpler	35	AU 36	Tongue Bulge
14	AU 15	Lip Corner Depressor	36	AU 37	Lip Wipe

15	AU 16	Lower Lip Depressor	37	AU 38	Nostril Dilator
16	AU 17	Chin Raiser	38	AU 39	Nostril Compressor
17	AU 18	Lip Puckerer	39	AU 40	Nasal Dilator - Flaring the nostrils
18	AU 19	Tongue Out	40	AU 41	Lid Droop - Drooping the eyelids
19	AU 20	Lip Stretcher	41	AU 42	Slit - Narrowing the palpebral fissure
20	AU 21	Neck Tightener	42	AU 43	Eyes Closure
21	AU 22	Lip Funneler	43	AU 45	Blink
22	AU 23	Lip Tightener	44	AU 46	Wink

2.3.2 Methods for Detecting the Facial Action Unit

In the Facial Action Coding System, facial action units (AUs) are the fundamental components of facial expressions. (FACS). Detecting AUs in facial expressions requires the identification of specific facial movements or alterations that are associated with the activation of specific facial muscles[70]. Here are some of the most common AU recognition techniques:

a. **Appearance-Based Techniques**: These methods use the texture and skin color tone on the face to analyze the appearance of the face in order to identify facial expressions. These methods identify facial traits and compare them with recognized AU patterns using image processing and feature extraction algorithms[70].

b. **Geometric-Based Techniques**: In order to identify facial expressions, these techniques examine the geometry and contour of the face. These methods track the movement of particular spots on the face using feature tracking algorithms in order to find AU patterns[70].

c. **Deep Learning-Based Techniques**: Deep neural networks are employed in these techniques to identify AU patterns from enormous datasets of annotated facial expressions. These methods have recently produced encouraging results and are gaining popularity in AU recognition research[71].

d. **Hybrid Techniques**: To increase the precision of AU recognition, these methods combine various strategies, such as appearance-based and geometric-based methodologies. Commercial AU recognition systems frequently employ hybrid strategies since they have shown results that are superior to those of individual techniques[72].

2.4 Facial Landmarks

Facial landmark detection is the process in computer vision that attempts to determine the prominent areas along the face, such as the position of the jaw, right and left brow, right and left eye, nose, and mouth. Facial landmark detection is a critical stage in facial analysis methods and many applications such as face swapping, facial expression recognition, face recognition, face alignment, emotion recognition, Facial action unit, and pose estimation[73][74]. Figure (2.3) location of facial landmark detection in the image.

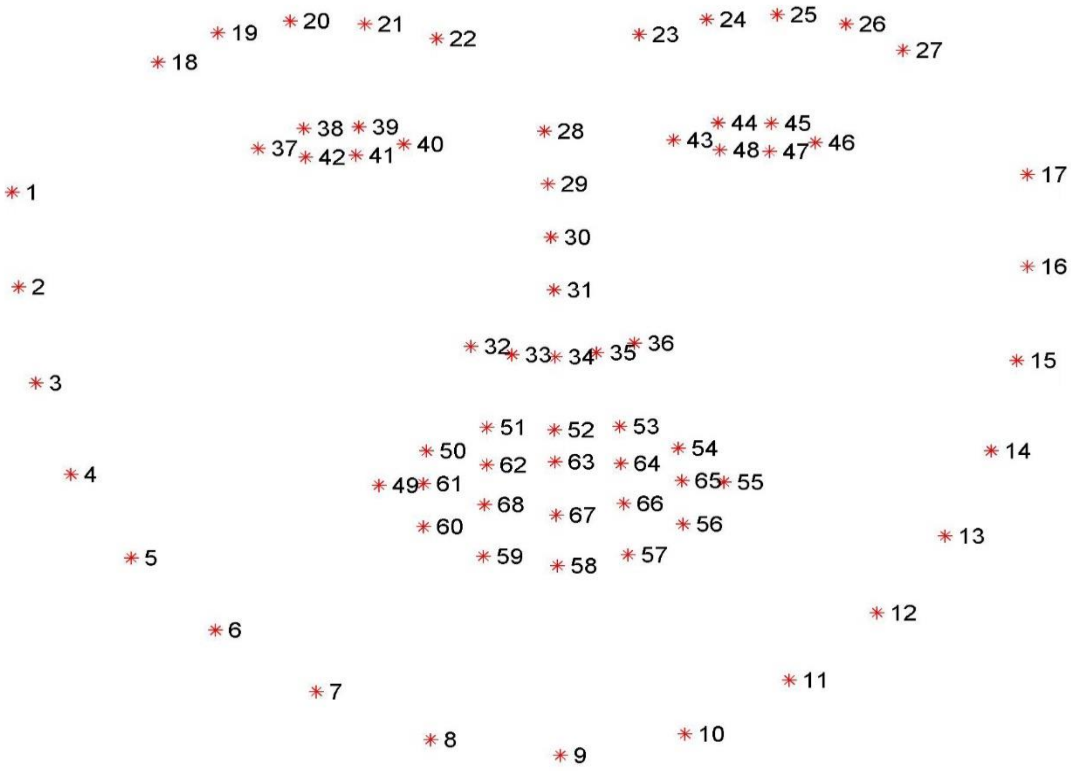


Figure (2.3): Location of The Facial Landmark[75]

2.4.1 Facial Landmarks Types

The face landmarks can be divided into two principal groups: primary and secondary, or fiducial and auxiliary, depending on how reliable the picture features are. One of the main categories of landmarks are those that provide broad information about the face, such as the corners of the eyes, the nose tip, the eyebrows, and the lips. These landmarks are easily identifiable using low-level image attributes. Applications for the fiduciary group include head tracking, face recognition, and facial detection[76]. The secondary group consists of a series of landmarks such cheek contours, eyelids, no extremity points on the midpoint of the eyebrows, etc. and it is frequently employed in applications where high accuracy in landmark identification is required, like facial emotion recognition[77][74]. Several primary and secondary landmark groups are shown in Figure (2.4).

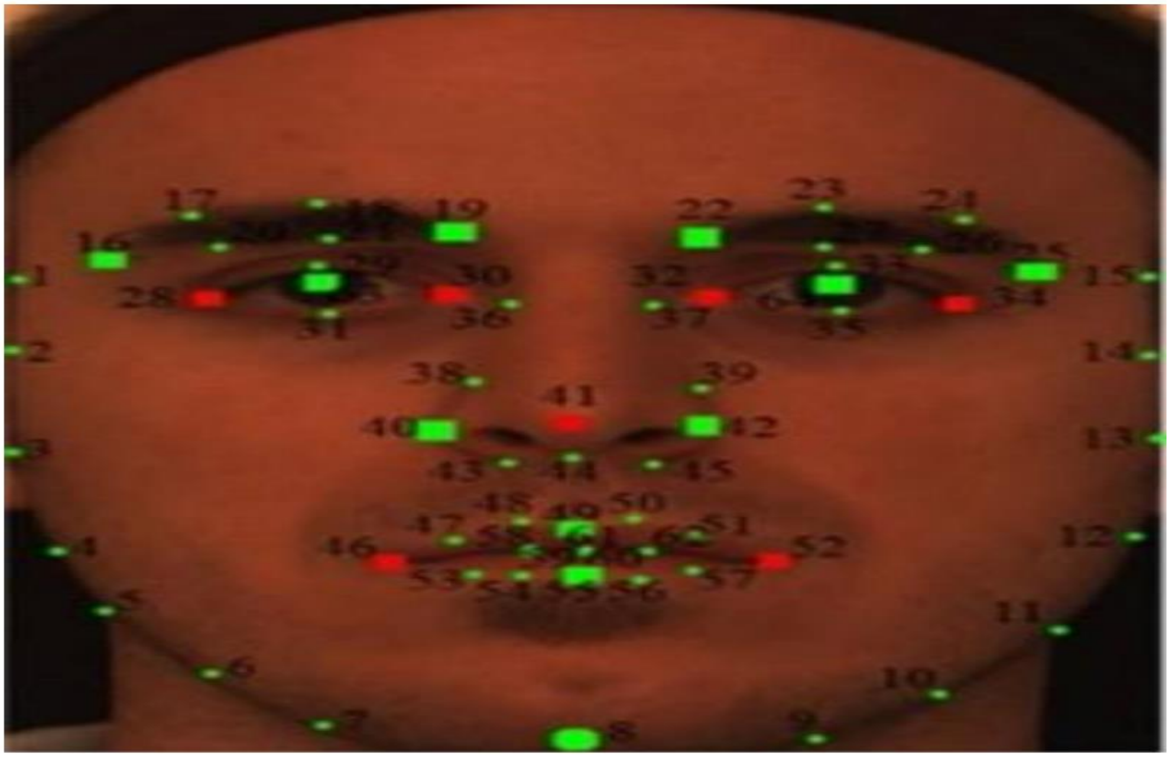


Figure (2.4): Groups of landmarks (the principal landmarks are shown as red squares, and the secondary landmarks are shown as green dots) [78]

2.4.2 Facial Landmark Detectors

There are numerous algorithms available for localizing landmarks with varying numbers of points. Each program selects the logical points needed for its objective[78][79]. The categories given in Figure (2.5) provide more clarity on the fundamental algorithms of facial landmark localization or detection.

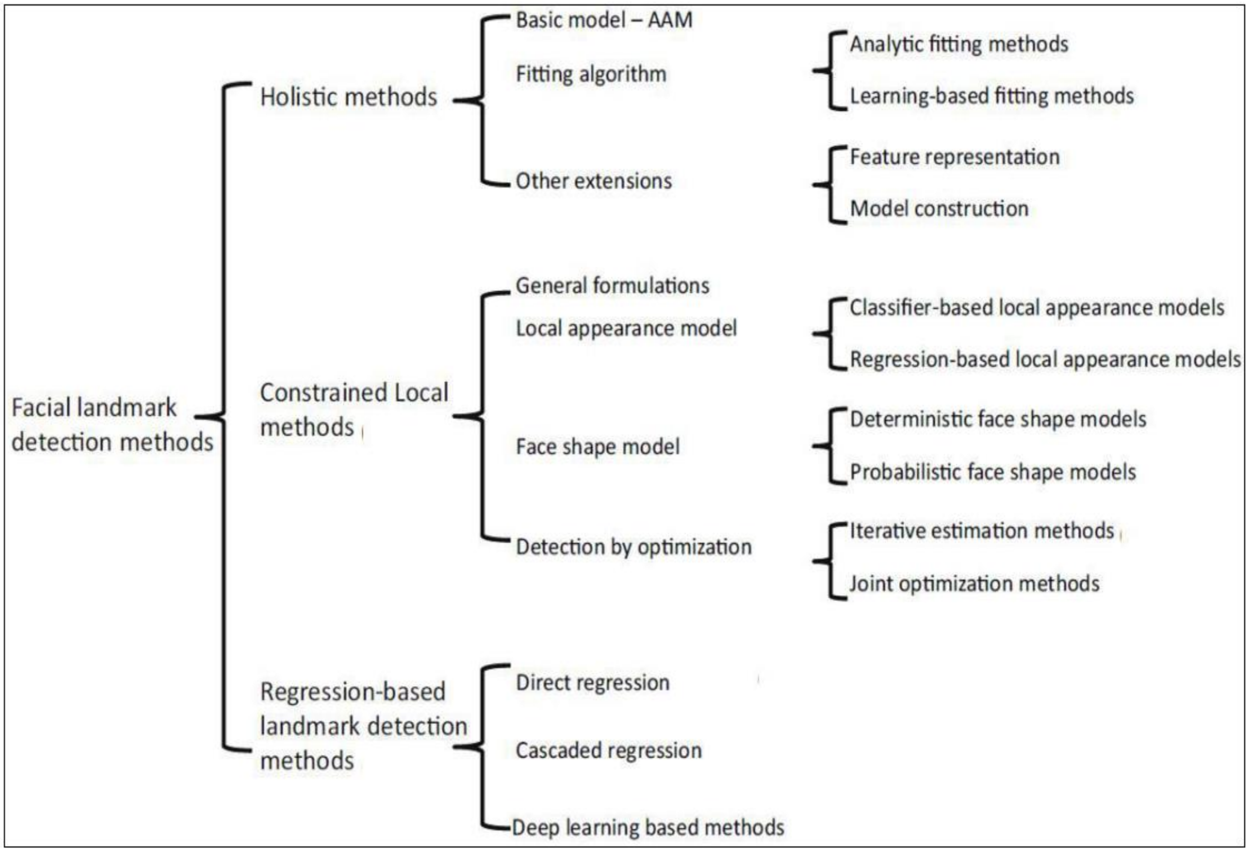

Facial landmark detection methods
Holistic methods
Basic model – AAM
Fitting algorithm
Analytic fitting methods
Learning-based fitting methods
Other extensions
Feature representation
Model construction
Constrained Local methods
General formulations
Local appearance model
Classifier-based local appearance models
Regression-based local appearance models
Face shape model
Deterministic face shape models
Probabilistic face shape models
Detection by optimization
Iterative estimation methods
Joint optimization methods
Regression-based landmark detection methods
Direct regression
Cascaded regression
Deep learning based methods

Figure (2.5): Algorithm Categories for Detecting Facial Landmarks[77]

2.5 Face Alignment

The process of transferring points from input coordinate systems (input images) to different output image coordinates is known as face alignment. For a vast range of applications, including Expression Analysis, Face Swap, Facial Aging Analysis, Facial Expression Analysis, and Face Verification [80], facial alignment is a crucial step. The face can be aligned using a variety of techniques, some of which try to apply a 3D model to the input image and then transform the image to fit that model[77][81]. Other simpler techniques rely on facial landmarks and geometric modifications. This technique creates a normalized rotation, translation, and scale representation of the face using the location of the eye area[80].

2.6 Face Pose Movement

The movement of a person's face in relation to their head and body is referred to as facial pose movement. It is concerned with the movement of facial features such as the eyes, nose, mouth, and brows, as well as the general form and location of the face. From happiness and excitement to sadness and rage, face pose movement can indicate a wide range of emotions and intents. A grin, for example, is characterized by an upward movement of the mouth and a rising of the cheeks, whereas a frown is characterized by a downward movement of the mouth and a furrowing of the brow[82]. The position and shape of the eyes, brows, and nose can also be altered by facial expressions. An astonished expression, for example, comprises widening of the eyes and raising of the brows, whereas a disgusted expression involves wrinkling of the nose and lowering of the brows. Facial landmark identification and tracking, which include detecting important spots on the face and measuring their position and movement over time, can be used to study and analyze face pose movement. This data can be utilized for a variety of purposes, such as facial identification, emotion recognition, and animation[83].

The head position in the image can be represented by an expression [84]:

$$(x_h, y_h, \psi, \theta, \phi)$$

Where the head's position in image coordinates is (X_h,Y_h).

ψ: is the roll of the rotation angles.

θ : is the pitch the rotation angles.

ϕ : is the yaw the rotation angles.

All of these rotation angles are constrained by $[-\frac{\pi}{2}, \frac{\pi}{2}]$, and their triple values for a frontal view of the head are [0, 0, 0]. The x-axis and y-axis rotation angles were denoted by the roll angle and pitch angle, respectively. The yaw angle was used to indicate the z-axis's counterclockwise rotation[77] [83]. Figure (2.6) can be used to depict the rotation angles. Consequently, the 3D head orientation matrix $R_{head} = R\psi R\theta R\phi$ is calculated as follows:

$$R_{\psi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & \sin\psi \\ 0 & -\sin\psi & \cos\psi \end{bmatrix}, R_{\theta} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}, R_{\phi} = \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

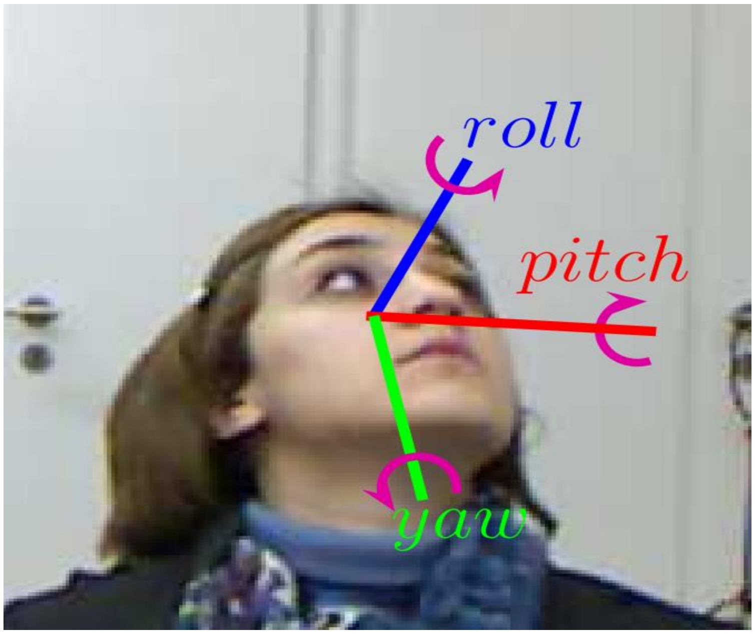


Figure (2.6): The Rotation Angles of Face Yaw, Pitch and Roll[82]

2.7 Deep Learning Approach

Deep learning is a subset of machine learning based on artificial neural networks that are supposed to learn hierarchically and generate predictions. The objective of deep learning is to develop neural networks that can learn from massive amounts of data and make accurate predictions or classifications based on that data[85].

In a deep learning technique, the neural network consists of numerous layers, each layer responsible for processing distinct input data characteristics [86]. The input data is sent to the first layer, which processes it and sends the output to the subsequent layer, and so on, until the final output is generated. Deep learning's capacity to automatically learn hierarchical data representations is one of its primary advantages. In image recognition tasks, for instance, lower network layers may learn to recognize simple elements such as edges and corners, whereas higher network layers may learn to recognize more complex features such as forms and patterns. This enables the network to recognize things and generate very accurate predictions[87].

A further benefit of deep learning is its capacity to manage massive amounts of data. Deep learning algorithms are meant to operate on vast datasets, enabling them to learn more complicated patterns and generate more accurate predictions[88] [89].

2.7.1 Types of Deep Learning

There are three main kinds of deep learning models. Each has its own network architecture and is used for something different and they are:

- **Supervised deep learning**

The model in supervised deep learning techniques is trained on pre-labeled data. The model is then used to adapt itself using the learning process, and during the testing phase, the model should select the correct answer without relying on any label. Supervised deep learning has two key applications: classification and regression issues. A convolution neural network is one of the most commonly used supervised models (CNN). AlexNet, LeNet-5, VGGNet, GoogleNet, ResNet, InceptionNet, and others are examples of CNN-based supervised deep learning architectures[90][91].

- **Unsupervised deep learning**

The model is trained on unlabeled data in unsupervised Deep Learning techniques, and the model attempts to extract patterns and features on its own. Deep Belief Network (DBN) and Restricted Boltzmann Machine (RBM) are two unsupervised deep learning architecture[91][92].

- **Deep Learning with Semi-Supervision**

Semi-Supervised Deep Learning is a step in between supervised and unsupervised learning. A huge amount of categorized data is required to support a greater collection of unlabeled data. This method is especially beneficial when extracting relevant data features is challenging and labeling samples takes a long time. This strategy is commonly used in Generated Adversarial Networks (GANs)[93][94].

2.7.2 Convolution neural network (CNN)

An artificial neural network called a convolutional neural network (CNN) process and analyzes data with a grid-like structure, like images or videos. CNNs are frequently used for image recognition, computer vision, and language processing tasks. The structure of CNNs is modeled after that of animal brain cortices. During the 1960s, CNN was the first successful architecture for deep learning because of the excellent training of the hierarchical layers. The CNN architecture takes advantage of spatial relationships to reduce the parameter number of the network, which improves performance when typical backpropagation algorithms are used. The growth of computation techniques and GPUs-accelerated have been used to train CNNs more effectively. The CNN performs two functions: the extraction function and the prediction function. Each function used certain layers to achieve a particular purpose [95][96].

2.7.2.1 Fundamental Elements of the CNN Architecture

The following five layers make up the majority of CNN's basic structure, in addition to functions for modifying and normalizing the data between the layers[97].

a. **The input layer**: The computer reads the data as an array, which can be used to predict what will happen in the end.

b. **Convolutional layer**: Every neuron in the convolutional layer is connected to neurons in the layers around it, and there are often many feature mappings. In a convolutional layer, there are often more than one feature map. Each feature map is made up of many shared weights that are set up in a rectangle pattern. During the training phase, these shared weights, called convolution kernels, will be changed often. The results of the convolutional layer will be fed into the next layer.

c. **The activation layer**: By making up for the shortcomings caused by the lack of a linear computational expression, the activation function is used to enhance the model's non-linear features.

d. **The pooling layer**: Maximum and average pooling are the most commonly used approaches in the pooling layer. Pooling can be thought of as a simplified version of convolution, or as a sampling layer.

e. **Flatten**: The depth of the convolution layers output may become greater than one. The flatten transforms the output of the convolution layers to create a flat structure that can then be fed into a fully connected layer.

f. **The fully connected layer (classification layer**): A multilayer classifier that is fully connected and is usually located at the network's end; this is where back-propagation will begin. When the forward propagation process reaches the final fully connected classification layer, weights and biases are updated to reduce losses and improve classification accuracy [97][98].

2.7.2.2 Architecture of a Two-Dimensional CNN

Two-Dimensional CNNs are built with the expectation that their input will be images. This section focuses on the architecture that should be implemented to handle the specific type of data [99]. Convolutional Neural Networks (CNN) are biologically inspired feed-forward networks in which the connections between neurons have a tendency to capture pattern invariances for the input data in the distortion or shift. The majority of CNN architectures assumed that the networks would operate on two-dimensional input data (usually images). A typical CNN is built from a series of layers, each of which transforms one volume of activations into another. Figure (2.7) depicts a CNN architecture that recognizes objects in an image[100][97].

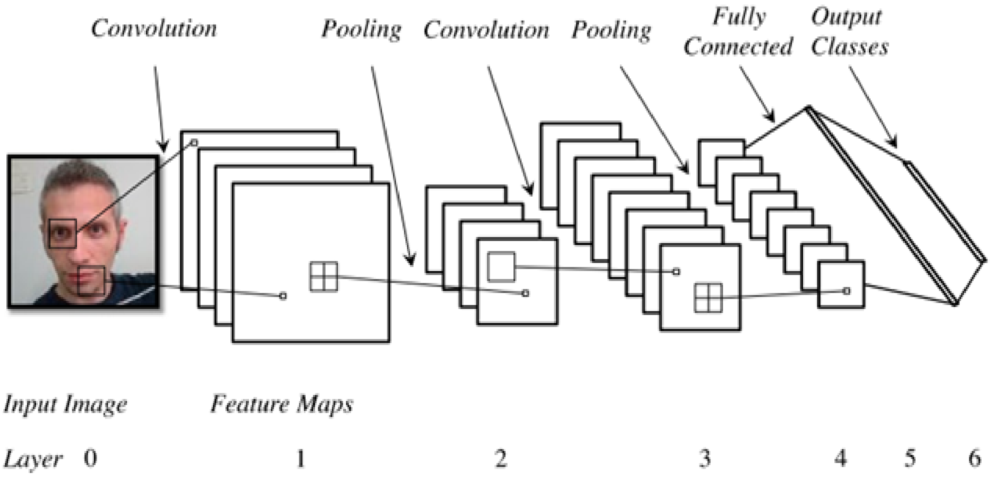


Figure (2.7): An Image Recognition Example of CNN Architecture [101]

a. Convolution Layer

In computer vision field concentrate on discrete 2D-convolution because it's the most common form of convolution in digital image processing. The convolution procedure determines the pixel intensity value in an image. To measure the corresponding pixel using the spatial dimensionality of the input (height and width) and the depth [102]. The depth refers to the third dimension of an activation volume. Neurons within a layer will only connect to a small portion of the layer preceding it. A convolution layer is a vital element of the CNN architecture that performs feature extraction. It is typically made up of a combination of nonlinear and linear processes such as convolution and activation functions. Convolution is a mathematical method for combining two groups of data. Convolution filters are used to extract features from the input image and learn these features using input data arrays, while feature maps are generated to cultivate the spatial relationship of each feature in the image[95][103].

The Convolution Layer works by sliding the filters over the input image, doing a dot product between the filter weights and the corresponding pixel values in the input image, and making a feature map that shows the parts of the input image that match the filter. The filter weights are learned during training, and they tell the convolution layer what kinds of features to pull out. In order to make the model less linear, the output of the convolution layer is often sent through an activation function like ReLU (Rectified Linear Unit). This is because images are not very linear, and the activation function lets the model capture complex relationships between the input image and the extracted features. The convolution layer is defined by a number of hyperparameters, such as the number of filters, their size, how far the sliding window moves, and how much the input image is padded. These hyperparameters control how big the output feature map is and how much detail the convolution layer can pull out of the input image[104].

In Figure (2.8) and Figure (2.9), for example, a 5 x 5 input image is convolved with a 3 x 3 kernel, and the output of the convolution operation is a 3 x 3 image matrix.

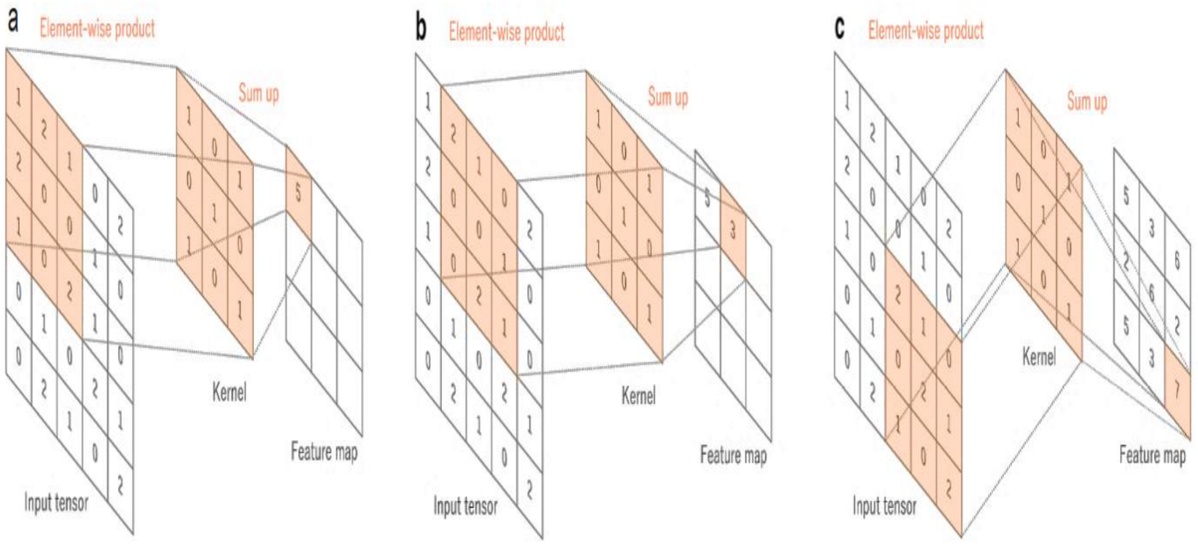
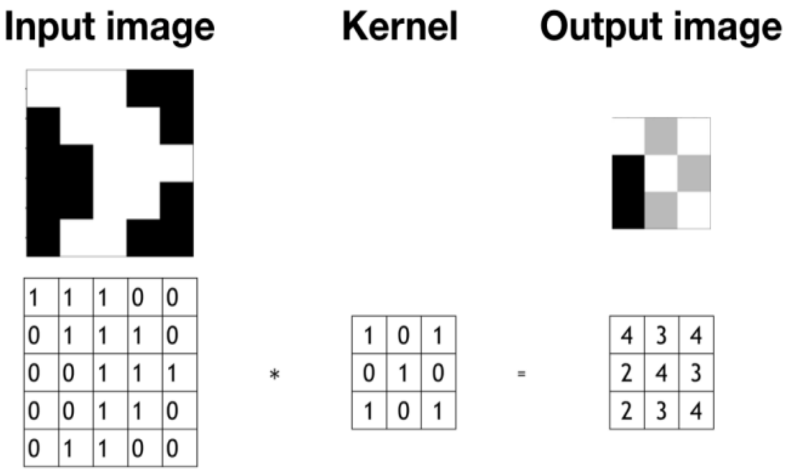


Figure (2.8) :Calculates the Convolution Operation [105]



$$\begin{bmatrix}1&1&1&0&0\\0&1&1&1&0\\0&0&1&1&1\\0&0&1&1&0\\0&1&1&0&0\end{bmatrix} * \begin{bmatrix}1&0&1\\0&1&0\\1&0&1\end{bmatrix} = \begin{bmatrix}4&3&4\\2&4&3\\2&3&4\end{bmatrix}$$

Figure(2.9) :Convolution Kernel Matrix Applied to Input and Output Images [106]

b. Separable Convolutions

CNNs have greatly increased the efficiency of machine learning and deep learning models. Nevertheless, it takes a long time to train such networks. Large datasets are required to train a high-performing model, resulting in extensive training times of up to several weeks. This is especially bad when evaluating network performance in order to make necessary changes. Nevertheless, even with the usage of virtual machines and significantly more efficient GPUs nowadays, training durations in machine learning projects continue to be an issue. So, separable convolutions fit this picture because they require less memory during training and have lower computational complexity; these networks are far more effective. They also outperform classic convolutions and offer a wider range of applications. This is because of the number of multiplications conducted throughout the training phase, which will be addressed further[107][102].

Convolutions that are depth-separable utilize kernels that can be factored into two smaller kernels. As a result, its popularity is growing. The depth wise separable convolution derives its name from the fact that it operates for both the spatial and depth dimensions of the channel count. An input image may contain three channels: RGB, which stands for red, green, and blue. Each channel represents a different interpretation of the image; for instance, the "red" channel represents the "redness" of each pixel, whereas the "blue" channel represents the "blueness" of each pixel, and so on [107]. Two steps comprise the depth wise separable convolution: a depthwise convolution and a pointwise convolution. In depthwise convolution, a spatial convolution is performed independently on each input channel, and then the outputs of the depthwise convolutions are mixed using pointwise convolution. In addition, as shown in Figure (2.10), convolution layers that are depth-separable can generate deeper information characters. Simultaneously, they require lower computing costs and a smaller number of parameters while delivering equivalent (or slightly improved) performance and scalability. Integrating many simple convolutions into a single block boosts the neural network's total depth and enables more accurate extraction of advanced features[108][88].

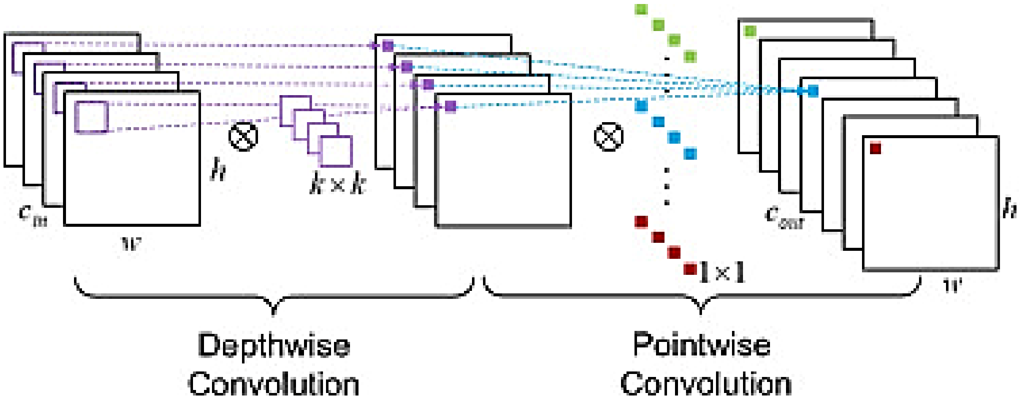


Figure (2.10): Architecture for Depth-Wise Separable Convolution[109]

The output size of the convolution layer is controlled by a number of hyperparameters that are part of the CNN architecture[110]. Here are some of the crucial CNN hyperparameters:

- **Filters**: A variety of filters in appropriate numbers and sizes can be employed.
- Filter size: The filter size or kernel size needs to be squared k*k, which is less than the input image.
- **Padding**: By filling at the input border, padding is a useful method for regulating the dimensionality of output volumes.
- **Stride**: the quantity of cells (pixels) the kernel must pass through or down the input map at once[109][111].
- **The activation layer (Non-Linear Layer)**: Rectified Linear Units (ReLUs) functions are just one of the many functions that non-linear layers might use. The distinguishing characteristics of every concealed layer are identified using these functions. In order to transform linear output into nonlinear, the feature maps output from the convolution layers are input to nonlinear layers [112][113].
- **Pooling layer** :The pooling layer changes the common feature representation into a more useful one that retains important information while removing irrelevant information, and in the pooling layer, the resolution of the feature maps is decreased in addition to improving the stability to deformation on the inputs[86]. Maximum pooling and average pooling are the two most used pooling strategies. Similar to how average pooling returns the average of the values of the image portion covered by the kernel[85], max pooling returns the highest value of the input map portion that the kernel covered. The Global Max Pooling layer and Global Average Pooling are further types of pooling layers.

The maximum value of the entire input map is determined to be the output in Global Max, whereas the average of all input map values is calculated to be the output value in Global Average Pooling [99][114]. In this case, the pool size is equal to the input size. As shown in Figure (2.11).

- **Optimizer**: The role of the optimizer in CNNs is to find the optimal set of weights and biases that allow the network to make accurate predictions on the training data and, importantly, generalize well to unseen data[112].

  Common optimizer algorithms used in CNNs and deep learning include:

  - **Stochastic Gradient Descent (SGD):** A fundamental optimizer that updates weights in the direction of the negative gradient of the loss function. Variants of SGD include mini-batch SGD, momentum, and Nesterov accelerated gradient[109][111].
  - **Adam**: A popular adaptive learning rate optimizer that combines the advantages of both Adagrad and RMSprop. It adjusts the learning rates for each parameter individually[112][113].
  - **RMSprop**: An optimizer that adapts the learning rate for each parameter based on the recent history of gradients[112][113].
  - **Adagrad:** An optimizer that adapts the learning rate for each parameter by scaling it inversely proportional to the square root of the sum of squared gradients[112][113].

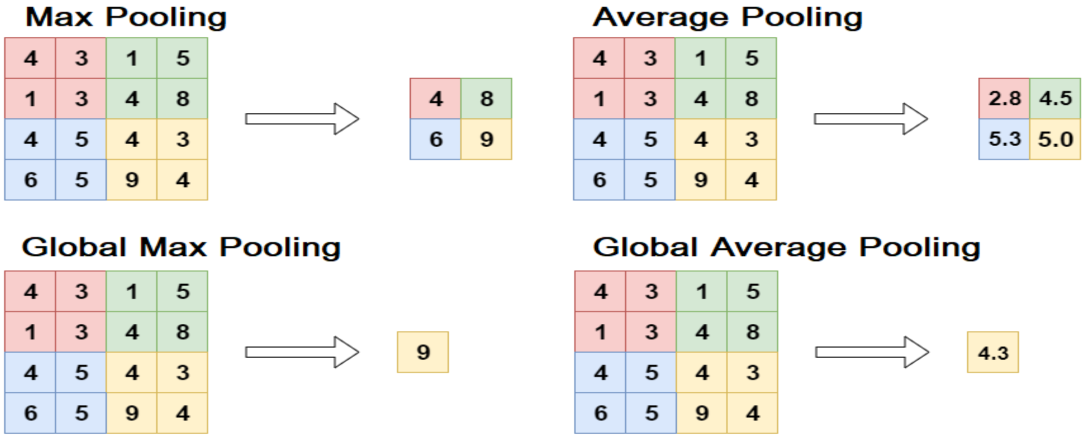
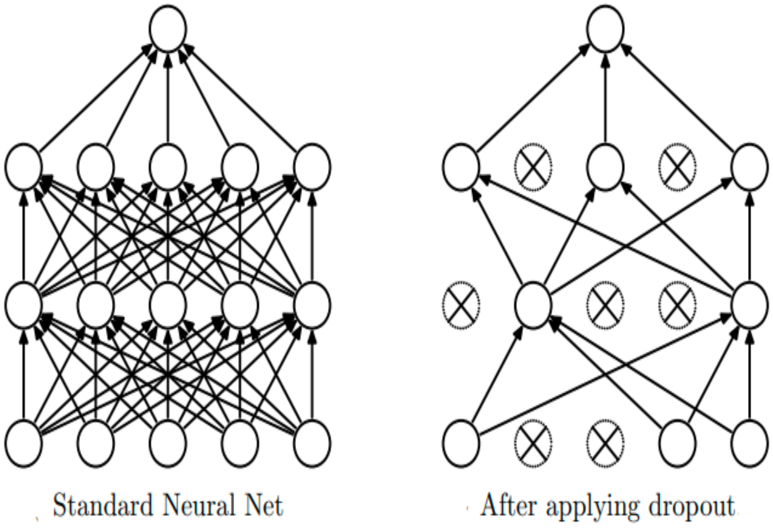


Figure (2.11): Pooling Layer Types [115]

- **Regularization layers:** One of the key problems with network models is overfitting. This problem arises in two situations: when the model is complex and when the dataset is inadequate. The model's accuracy will be high during the training phase and low during the testing phase because overfitting makes it harder to generalize the model to previously unknown samples. One of the key strategies for preventing the overfitting issue is to use regularization layers[86][116]. The two regularization layers that are most frequently employed are represented as follows:

  - Layer dropout

  A method for dealing with overfitting is dropout. The term "dropout" describes the removal of units, both visible and hidden. Dropping a unit from a neural network is deleting it logically, or in other words, cutting off its incoming and outgoing connections. These units are selected at random based on value, as shown in Figure (2.12) where Example can be set at 0.5[100][85].



Figure(2.12): The Dropout Process [117]

- Activating the batch normalization layer

For steady generalization improvement and optimization, deep networks require layers of normalization and activation functions. One method for regularizing the neural network and preventing overfitting is batch normalization. The distributions of the inputs to the current layer vary as a result of changes in the parameters of the preceding layer, which is known as the Internal Covariate Shift issue. To improve training performance, the current layer must constantly be adjusted to the new distribution. Fixing the output of the preceding layers to have a mean of (0)and a standard deviation of (1) normalizes the result[118][105].

The network grows increasingly clever to recognize shapes that are more challenging than those identified by the preceding layer when training is continued at each layer. When the novice layers are able to distinguish between colors, lines, corners, etc. Finally, the deeper layers may distinguish between shape elements and overall shapes[119]. Figure (2.13) is a representation of CNN layers that can provide clarification.

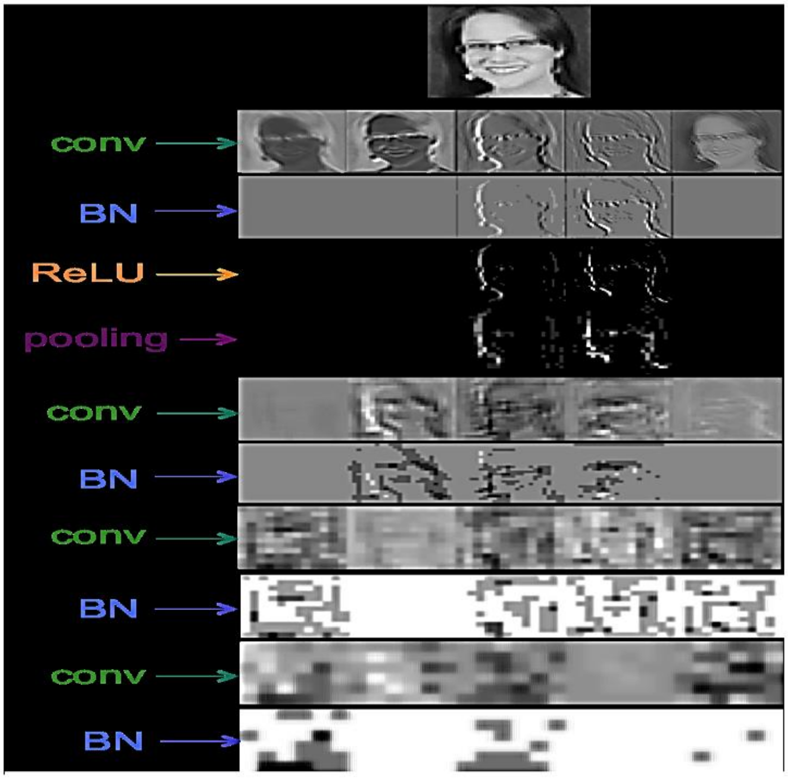


Figure (2.13) : CNN Layers Visualization [120]

2.7.3 Recurrent Neural Network (RNN)

A Recurrent Neural Network (RNN) is a type of deep learning architecture designed for processing sequences of data. Unlike traditional feedforward neural networks where data flows in a one-way direction, RNNs have connections that loop back on themselves, allowing RNNs to maintain a form of memory and capture temporal dependencies in sequential data[121]. Here's a breakdown of the key concepts in an RNN:

a. **Sequential Data:** RNNs are particularly well-suited for processing sequential data, where the order of elements matters. This includes time series data, natural language text, speech signals, and more[122].

b. **Recurrent Connections**: The fundamental feature of an RNN is its recurrent connections, which allow information to be passed from one step of the sequence to the next. This enables the network to maintain a form of memory about previous steps[123].

c. **Hidden State:** At each time step, an RNN maintains a hidden state vector, which serves as both input and memory for the current step. The hidden state is updated based on the input at the current step and the hidden state from the previous step[124].

d. **Weight Sharing:** In an RNN, the same set of weights and biases are applied at every time step, allowing the network to share information across different steps of the sequence[124].

e. **Vanishing Gradient Problem:** One challenge with traditional RNNs is the vanishing gradient problem. When training deep networks over long sequences, gradients can become very small, making it difficult to learn long-range dependencies. This can result in poor performance for long sequences[125].

2.7.3.1 Gated Recurrent Unit (GRU)

GRU is a type of RNN that is often used in natural language processing (NLP), speech recognition, and other tasks that involve moving from one step to the next. Cho et al. came up with it in 2014 as a simpler and more efficient option to the LSTM (Long Short-Term Memory) network that could still recognize long-term dependencies in sequences. The restart gate and the update gate are the most important parts of GRU. These gates decide how much information from the last time step gets passed on and how much new information gets added to the current time step. The reset gate is responsible for deciding how much of the previous hidden state (h_t-1) to forget and how much of it to remember. It takes as input the concatenation of the previous hidden state and the current input (x_t), and passes it through a sigmoid function, as shown in Equation (2.2)[126][127].

$$r_t = \sigma(W_r[h_t - 1, x_t] + b_r) \dots\dots (2.2)$$

where W_r and b_r are learnable weight and bias parameters, and σ is the sigmoid function. The update gate determines how much of the previous hidden state to keep and how much of the new candidate hidden state ($\tilde{h}_t$) to incorporate into the current hidden state. It takes as input the concatenation of the previous hidden state and the current input (x_t), and passes it through a sigmoid function, as shown in Equation (2.3)[128].

$$z_t = \sigma(W_z[h_t - 1, x_t] + b_z) \dots\dots\dots\dots\dots\dots\dots\dots..\dots (2.3)$$

where W_z and b_z are learnable weight and bias parameters.

Applying a hyperbolic tangent (tanh) activation function to the concatenation of the previous hidden state that was reset-gated and the current input yields the candidate hidden state, as shown in Equation (2.4)[128].

$$\tilde{h}t = tanh(Wh\,[rt \odot h_t - 1, xt] + bh) \dots\dots (2.4)$$

where Wh and bh are learnable weight and bias parameters, and ⊙ denotes element-wise multiplication.

Finally, the current hidden state is computed by combining the previous hidden state and the candidate hidden state based on the update gate, as shown in Equation (2.5)[128].

$$h_t = z_t \odot h_t - 1 + (1 - z_t) \odot \tilde{h}_t \ldots\ldots\ldots\ldots\ldots\ldots (2.5)$$

where ⊙ denotes element-wise multiplication.

2.7.3.2 Long Short-Term Memory (LSTM)

LSTM is a form of RNN that is frequently employed in tasks involving sequential input, such as speech recognition and natural language processing. Traditional RNNs can experience the vanishing gradient problem, which makes it challenging for the network to learn long-term dependencies in the data[129]. LSTM networks are made to overcome this issue. A mixture of gates and memory cells is used by LSTM networks to selectively remember or forget data from the previous time step. The sigmoid activation functions that control the gates output values between 0 and 1, which specify how much information should be allowed through. The data is kept in the memory cells and can be updated or retrieved as required. Input gate, forget gate, output gate, and memory cell are the four fundamental parts that make up an network's equations[129]. The values for the gates and memory cell at each time step are determined using these formulae[130].

Here are the equations for an LSTM network:

- Input gate :

$$i\_t = \sigma(W\_i[x\_t, h\_\{t - 1\}] + b\_i) \ldots\ldots (2.6)$$

where **i_t** is the input gate vector, **W_i** is the weight matrix for the input gate, **x_t** is the input vector at time **t**, **h_{t-1}** is the hidden state vector from the previous time step, and **b_i** is the bias vector for the input gate. The **σ** function is the sigmoid activation function[130].

- Forget gate :

$$f\_t = \sigma(W\_f[x\_t, h\_\{t-1\}] + b\_f)\,) \ldots\ldots (2.7)$$

where **f_t** is the forget gate vector, **W_f** is the weight matrix for the forget gate, **x_t** is the input vector at time **t**, **h_{t-1}** is the hidden state vector from the previous time step, and **b_f** is the bias vector for the forget gate[130].

- Memory cell:

$$c\_t = f\_t * c\_\{t-1\} + i\_t * tanh(W\_c[x\_t, h\_\{t-1\}] + b\_c)) \ldots\ldots (2.8)$$

where **c_t** is the memory cell vector, **f_t** is the forget gate vector, **c_{t-1}** is the previous memory cell vector, **i_t** is the input gate vector, **W_c** is the weight matrix for the memory cell, **x_t** is the input vector at time **t**, **h_{t-1}** is the hidden state vector from the previous time step, **b_c** is the bias vector for the memory cell, and **tanh** is the hyperbolic tangent activation function[130].

- Output gate:

$$o\_t = \sigma(W\_o[x\_t, h\_\{t-1\}] + b\_o) \ldots\ldots (2.9)$$

where **o_t** is the output gate vector, **W_o** is the weight matrix for the output gate, **x_t** is the input vector at time **t**, **h_{t-1}** is the hidden state vector from the previous time step, and **b_o** is the bias vector for the output gate[130].

- Hidden state:

$$h\_t = o\_t * tanh(c\_t) \ldots\ldots (2.10)$$

where **h_t** is the hidden state vector at time **t**, **o_t** is the output gate vector, **c_t** is the memory cell vector, and **tanh** is the hyperbolic tangent activation function.

In conclusion, the input gate chooses how much new data should be stored in the memory cell, the forget gate chooses how much old data should be kept in the memory cell, and the memory cell updates and saves the data[130][131].

2.7.3.3 Bidirectional Long Short-Term Memory (BiLSTM)

A modification of the Long Short-Term Memory (LSTM) paradigm that permits bidirectional processing of input sequences is known as Bidirectional Long Short-Term Memory (BiLSTM). It is frequently employed in natural language processing (NLP) jobs where understanding the context of a sentence's words is crucial as shown in Figure (2.14) [129]. A forward LSTM network and a backward LSTM network make up the BiLSTM model. The input sequence is processed in the forward direction by the forward LSTM and in the reverse direction by the backward LSTM. The final output is created by concatenating the results of two LSTMs[132]. In a BiLSTM model, the forward and backward LSTMs' mathematical equations are as follows:

In the Equation (2.11), the Equation (2.12), and the Equation (2.13), BiLSTM expression is displayed[133].

$$\vec{h}_R^{\square} = \sigma(w_{\overrightarrow{xh}} x_r + w_{\vec{h}\vec{h}} \overrightarrow{h_{r-1}} + b_{\vec{h}}) \quad \text{.........(2.11)}$$

$$\overleftarrow{h}_R^{\square} = \sigma(w_{\overleftarrow{xh}} x_r + w_{\overleftarrow{h}\overleftarrow{h}} \overleftarrow{h_{r-1}} + b_{\overleftarrow{h}}) \quad \text{.........(2.12)}$$

$$H_R^{\square} = \sigma(w_{\overrightarrow{xh}} \vec{h} + w_{\overleftarrow{xh}} \overleftarrow{h} + b_y) \quad \text{.........(2.13)}$$

When h stands for the hidden state, x stands for the vector of features, and σ is the logistic sigmoid function. The three gates' weight matrices (W terms) and bias vectors

(B terms), respectively, are represented as W terms and b terms. the output is produced by updating both the forward $\vec{h}$ and backward $\overleftarrow{h}$ structures. For NLP applications that call for context data from both sides of an input sequence, the BiLSTM model is a potent tool. Numerous applications, such as sentiment analysis, named entity recognition, machine translation, and speech recognition, have made use of it[133].

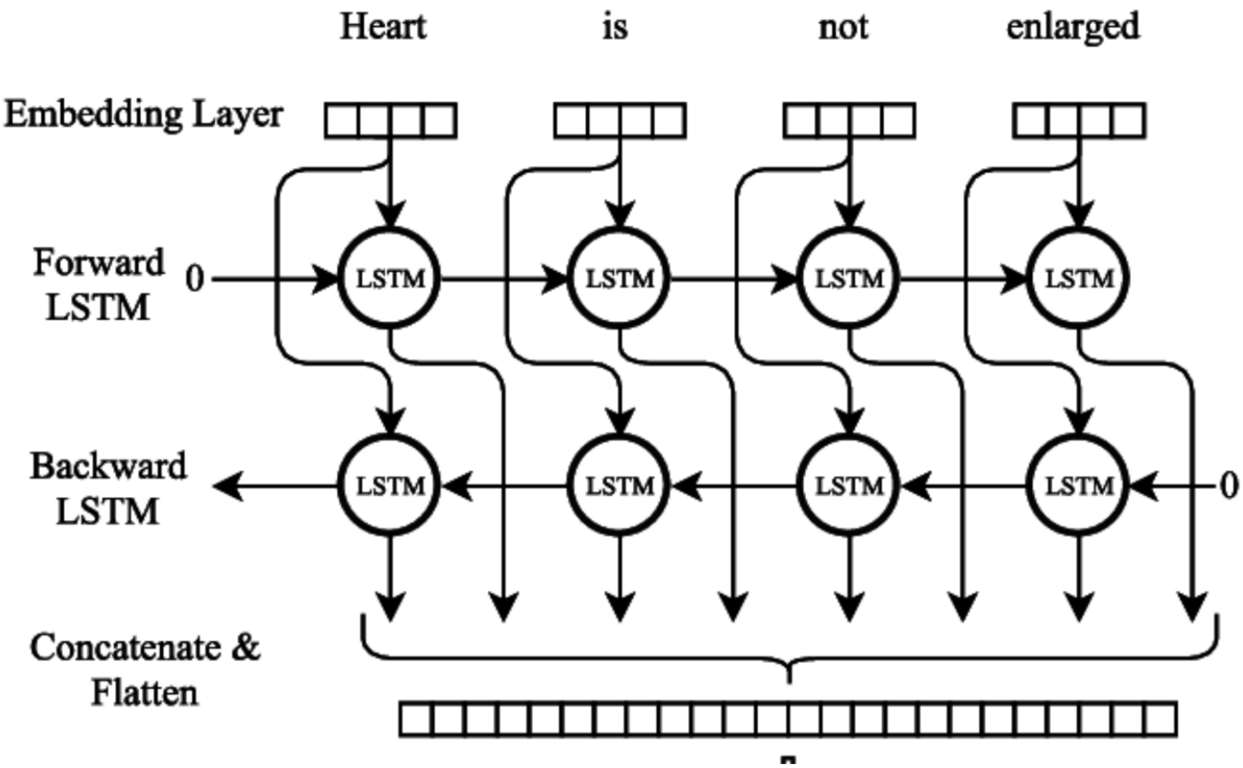


Figure (2.14): The BiLSTM Model for NLP Applications[134]

2.8 Transfer Learning

Transfer learning is a type of machine learning in which the knowledge gained from performing one task is used to improve the performance of another task that is similar but different. Transfer learning is the process of fine-tuning or adapting a model that has already been trained on a large dataset to do a certain job for a new task. This method works well when the new task doesn't have a lot of information or when the tasks have some similar connections[135]. In deep learning, transfer learning usually refers to using pre-trained neural network models as a starting point for a new task. Pre-trained models are neural network architectures that have been trained on large datasets to do a certain task, like recognizing images or understanding natural language. These models are trained on powerful hardware and huge datasets, which makes them able to find complicated patterns in the data[136].

2.8.1 JAA-Net

JAA-Net: Joint Facial Action Unit Detection and Face Alignment via Adaptive Attention is a deep learning architecture that is used for image super-resolution and other computer vision tasks. This expansion focuses on facial action unit (AU) recognition and face alignment tasks, which are useful in areas such as human-computer interaction, affective computing, and facial expression analysis. The design is made up of two primary parts: an autoencoder and an attention-based neural network with adaptive attention. The autoencoder extracts high-level features from facial photos, which are subsequently fed into the attention-based neural network. For both AU detection and face alignment tasks, the attention-based neural network employs an adaptive attention strategy to focus on the most essential regions of the face. By preferentially focusing on the most relevant regions of the face, the adaptive attention mechanism dynamically adjusts its focus based on the input data, which can increase the accuracy of both tasks. Joint Facial Action Unit Detection and Face Alignment via Adaptive Attention beats existing state-of-the-art approaches for both tasks, according to experimental results on the BP4D and DISFA datasets. This method could be useful in a variety of disciplines where facial expression analysis is crucial, such as affective computing, human-computer interaction, and healthcare[137].

The JAA-Net framework is shown in Figure (2.15), and it consists of four modules (each represented by a different color): face alignment, global feature learning, hierarchical and multi-scale area learning, and adaptive attention learning. First, JAA-Net is built on a hierarchical and multi-scale area learning framework that captures information from each local region at several scales. Second, the face alignment module is made to make estimates of the positions of facial landmarks, which will then be used to produce the preliminary attention maps for AU detection. The purpose of the global feature learning module is to record the shape and texture characteristics of the entire face. The next step is adaptive attention learning, which uses a multi-branch network to learn the attention map of each AU adaptively in order to collect local AU information at various locations. The layers of the hierarchical and multi-scale region learning are shared by the three modules face alignment, global feature learning, and adaptive attention learning which are cooperatively maximized[137].

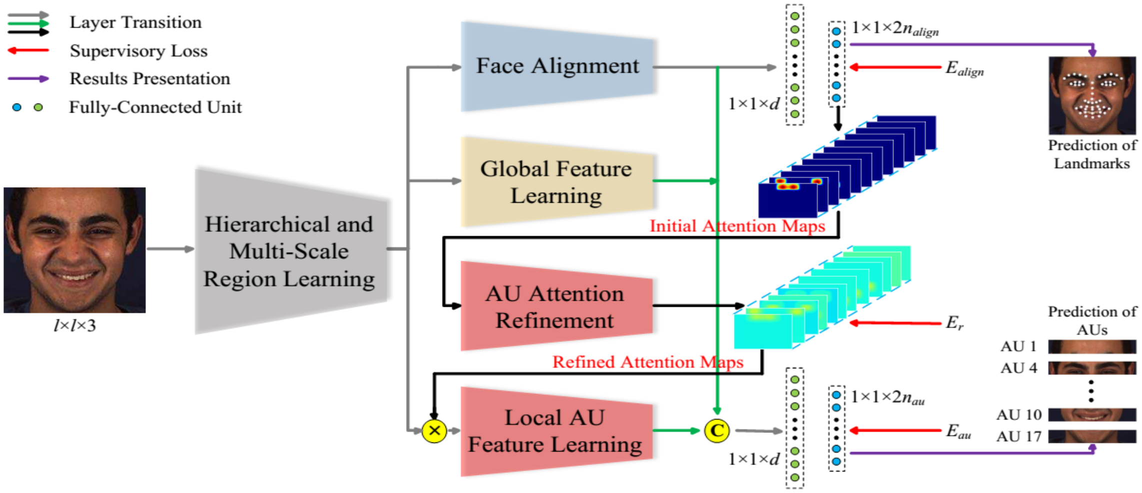


Figure (2.15): The JAA-Net Framework, where " × " and "C" Stand for Element-Wise Multiplication and Concatenation, Respectively[137]

JAA-Net seeks to accomplish AU detection and face alignment simultaneously. It uses a color face of $l \times l \times 3$ as input, and adaptively refines the attention mappings of AUs. JAA-Net's overall loss is what it refer to as Equation (2.14).

$$E = E_{au} + \lambda_1 E_{align} + \lambda_2 E_r \;\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.14)$$

where E_r measures the difference between before and after the attention refinement, which is a restriction to maintain the consistency, and $E_{au}$and E_{align} signify the losses of AU detection and face alignment, respectively. λ_1 and λ_2 are trade-off parameters. The hierarchical and multi-scale region layer to learn characteristics of each local region with various scales, as shown in Figure (2.16). One convolutional layer and three further hierarchical convolutional layers with variously sized weight-sharing areas are contained in a block of hierarchical and multiscale region layer[137][72].

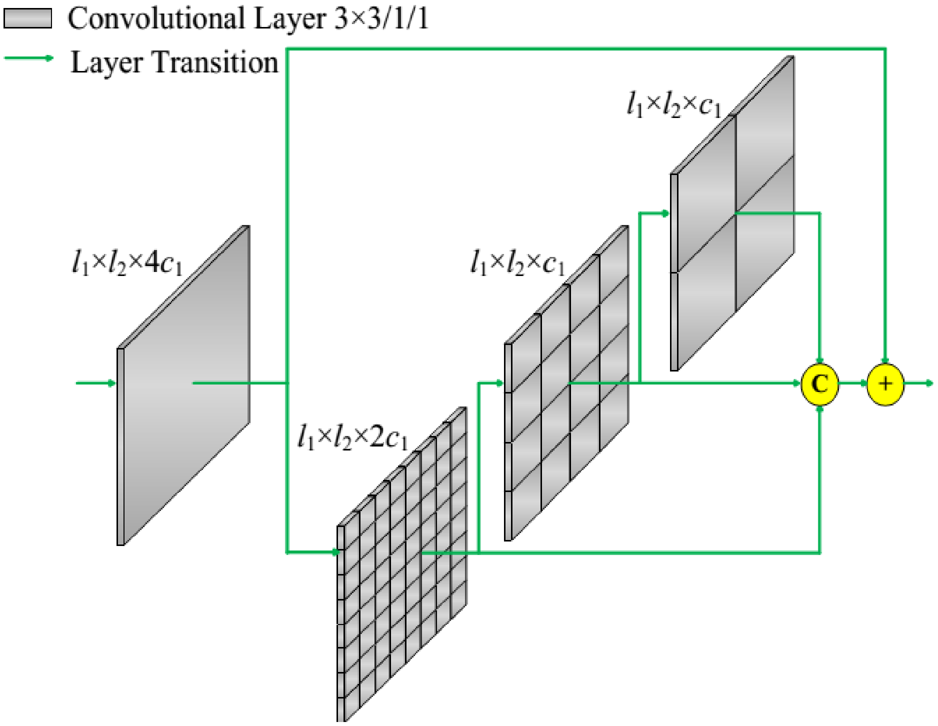

Convolutional Layer 3×3/1/1
Layer Transition
$l_1 \times l_2 \times 4c_1$
$l_1 \times l_2 \times 2c_1$
$l_1 \times l_2 \times c_1$
$l_1 \times l_2 \times c_1$
C
+

Figure (2.16): The Hierarchical And Multi-Scale Region Layer where" × " and "C" Stand for Element-Wise Multiplication and Concatenation, Respectively[72]

The second module is the face alignment module consists of three convolutional layers. The output of this module is fed into a landmark prediction network consisting of two fully connected layers of dimension d and $\mathrm{N}_{\mathrm{align}}$, where $\mathrm{N}_{\mathrm{align}}$ is the number of face landmarks. It defines the loss of face alignment, as shown in Equation (2.15) [72].

$$E_{align} = \frac{1}{2d_0^2}\sum_{j=1}^{\mathrm{N}_{align}} [\left(y_{2j-1} - \hat{y}_{2j-1}\right)^2 + (y_{2j} - \hat{y}_{2j})^2] \ldots\ldots\ldots.(2.15)$$

where y_{2j-1} and y_{2j} represent the ground-truth x-coordinate and y-coordinate of the j-th facial landmark, $\hat{y}_{2j-1}$ and $\hat{y}_{2j}$represent the corresponding anticipated outcomes, and d_0 represents the ground-truth inter-ocular distance for normalization.

The architecture of the adaptive attention learning is depicted in Figure (2.17) [137]. It consists of two steps: learning local AU features and AU attention refinement, where the first step entails improving the attention map of a particular AU with a branch and the second step entails learning and extracting local AU features. Initialization and refined attention map findings, respectively, serve as the inputs and outputs of the AU attention refinement process. The attention distributions of the predefined ROI and remaining areas are both fine-tuned in each AU's attention map, which has the size $l/4 \times l/4 \times l$ [138].

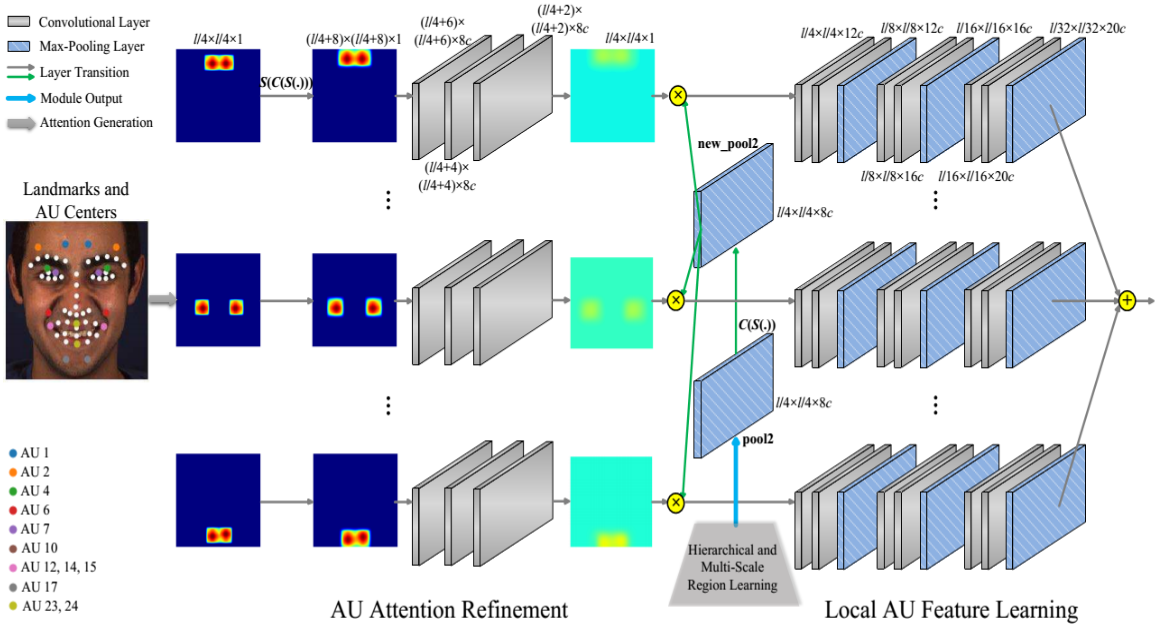


Figure (2.17): The Adaptive Attention Learning Architecture where" × " and "+" Stand for Element-Wise Multiplication and Sum Operations, Respectively[137]

Due to the symmetry, the designated ROI of each AU has two AU centers, each of which is the epicenter of a subregion. Particularly, the predicted face landmarks serve as preset sites for AU centers. If the k-th point of the attention map for the i-th AU is in a portion of the predetermined ROI, its attention weight is initially, as shown in Equation (2.15) [138].

$$v_{ik} = \max\left\{1 - \frac{d_{ik}\xi}{(l/4)\ \ \xi}, o\right\}, \qquad i = 1, \dots, n_{au} \ \dots\dots\dots.(2.16)$$

where d_{ik} is the Manhattan distance from this point to the AU center of the subregion, ζ is the ratio between the width of the subregion and the attention map, $\xi \geq 0$ is a coefficient, and n_{au} is the number of AUs[72].

The JAA-Net network is trained on Dataset Bp4D. A 3D video database of spontaneous facial expressions from a variety of young individuals is called the BP4D-Spontaneous dataset. Emotional and paralinguistic communication were elicited using well-validated emotion inductions. Using the Facial Action Coding System, frame-level ground truth for facial actions was determined. In both 2D and 3D domains, facial features were monitored using both person-specific and general methods. Forty-one participants are included in the database (23 women and 18 men). They ranged in age from 18 to 29; 20 of them were Euro-American, 11 were Asian, and 6 were African-American. To successfully elicit participants' emotions, an emotion elicitation methodology was created. At this stage, the JAA-Net network is trained and becomes able to predict the AU[72].

2.8.2 Feature Pyramid Network (FPN)

FPN is a design for a deep neural network that tries to solve the problem of detecting and recognizing objects at different scales. The idea behind FPN is to take feature maps from different steps of a convolutional neural network (CNN) that have different spatial resolutions and then combine them into a pyramid shape[139]. A bottom-up pathway and a top-down route make up the FPN architecture. The bottom-up path starts with an image as input and uses a number of convolutional layers to pull out feature maps at different levels of abstraction. The top-down path then takes these feature maps and, in a process called "lateral connections," makes them more detailed. The feature maps that come out of both the bottom-up and the top-down paths are then combined to make a feature pyramid[140]. The feature pyramid makes it possible to find and recognize items at different sizes, since different-sized objects may be better shown in different-resolution feature maps. FPN has been used in a number of computer vision tasks, such as object recognition, semantic segmentation, and instance segmentation, where it has performed at or near the top of the field[141].

2.8.3 FASTER R-CNN (Region-based Convolutional Neural Network)

The state-of-the-art object identification model FASTER R-CNN expands on the R-CNN and Fast R-CNN designs. Localizing and detecting things inside an image is a fundamental job in computer vision known as object detection. FASTER R-CNN offers an accurate and effective end-to-end object detection solution[83].

A Region Proposal Network (RPN) and a Fast R-CNN detector are the two major parts of FASTER R-CNN. Together, these elements create region recommendations, which are subsequently categorized and improved in order to identify objects in the image, as shown in Figure (2.18) [142].

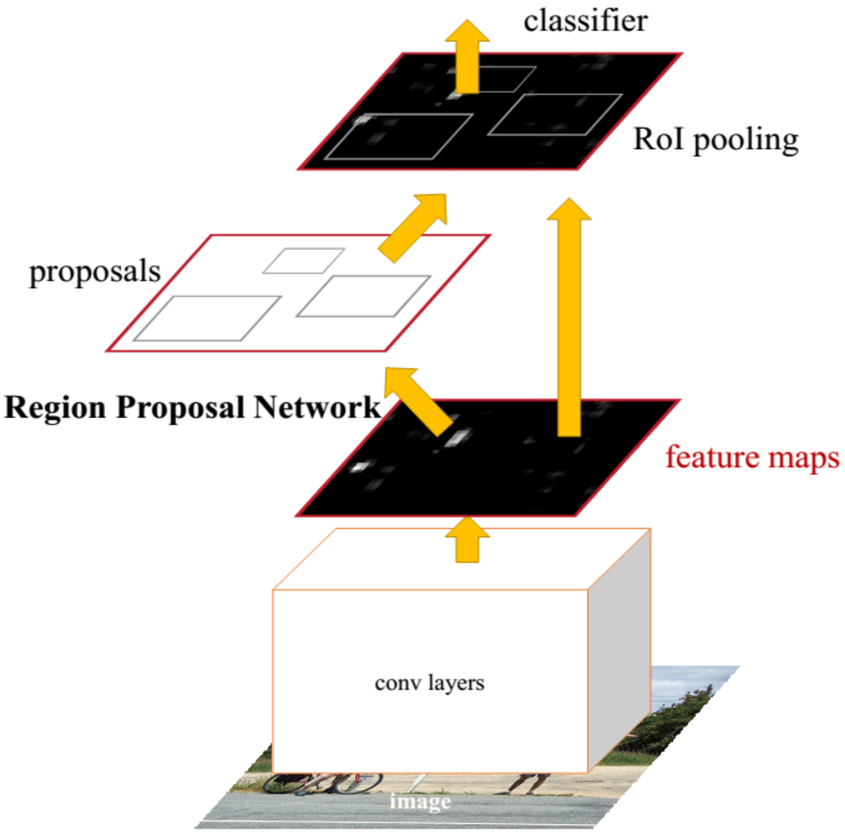


Figure (2.18): Faster R-CNN Network[142]

RPN, or Region Proposal Network, which are candidate object bounding boxes that may include objects, are created by the RPN. The RPN generates ideas at various scales and aspect ratios using a sliding window method. The Fast R-CNN detector shares convolutional layers with the RPN, a fully convolutional network. The RPN creates a series of rectangular proposals that could contain objects using the feature map of the input image. Each proposal has a probability score attached to it that represents how likely it is that an object will be present[142].

The RPN first applies a convolutional layer to the input picture to create a series of feature maps before producing region recommendations. Then, using a series of region proposal networks (RPNs), a set of candidate object boxes are predicted for each of the feature maps, along with the corresponding objectness scores. The RPN generates proposals at various scales and aspect ratios using a sliding window technique, and then rates each proposal according to how likely it is to contain an object. The Fast R-CNN detector is then given access to the proposals with the highest scores[143].

The Fast R-CNN detector uses a CNN to extract features from each region proposal after receiving input from the RPN's region proposals. The class and specific bounding box placement of the object within the proposal are then predicted using these attributes, which are subsequently given into a classifier and a regressor[144].

The Fast R-CNN detector uses RoI (Region of Interest) pooling to extract a fixed-size feature vector from the feature maps using a region proposal created by the RPN. In order to create a fixed-length feature vector, the RoI pooling process splits the proposal region into a predetermined number of sub-windows and applies max pooling within each one. Two distinct fully connected layers; one for classification and the other for bounding box regression, which are then supplied with the generated feature vector. While the bounding box regression layer fine-tunes the bounding box coordinates of the object within the proposal, the classification layer forecasts the likelihood that the object will belong to each class. The detector's ultimate output is a collection of object classes and the bounding boxes that go with them[144][145].

One of FASTER R-CNN's main advantages over its forerunners is its end-to-end training procedure, which enables the model to simultaneously master the region proposal generation and object identification tasks. As a result, object detection performance is improved and is faster, hence the term "FASTER R-CNN". Combining classification loss with bounding box regression loss, the model is trained. The bounding box regression loss penalizes the model for inaccurate bounding box predictions, whereas the classification loss penalizes the model for inaccurate class predictions. The losses are merged into a single multi-task loss function, and stochastic gradient descent is used to optimize it[145].

2.8.4 ResNet-18

ResNet-18 is a convolutional neural network architecture introduced by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun in their 2015 paper "Deep Residual Learning for Image Recognition." It was created to combat the issue of vanishing gradients in extremely deep neural networks, which can hinder their capacity to learn. ResNet-18 consists of 18 layers of residual nodes. A residual block is comprised of two or more convolutional layers with a connection bypassing one or more layers. This connection enables the network to learn residual functions, which are the differences between a layer's input and output. Using residual functions, the network can learn to modify the output of a layer without altering its input, thereby preventing the problem of vanishing gradients[146].

ResNet-18's architecture can be divided into four phases as shown in Figure (2.19). A solitary convolutional layer is followed by a max-pooling layer in the first stage. Each residual block in the second stage has two convolutional layers. Each residual block in the third stage consists of three convolutional layers. The fourth and final stage consists of two residual blocks with four convolutional layers each. Each residual block in ResNet-18 has a set of hyperparameters that can be adjusted to optimize the efficacy of the network. These hyperparameters include the number of filters in each convolutional layer, filter size, convolution stride, and buffering. ResNet-18 has attained state-of-the-art performance on a variety of computer vision tasks, including image classification, object detection, and semantic segmentation, after being trained on large datasets such as ImageNet. It is computationally effective and well-suited for applications with limited computing resources due to its relatively shallow depth[147].

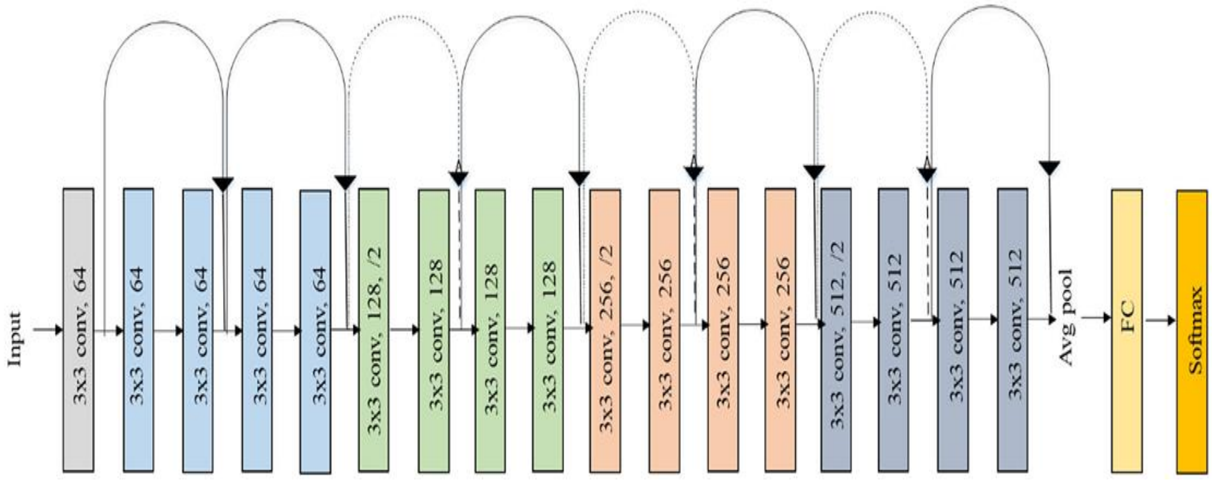


Figure (2.19): ResNet-18 Architecture[148]

2.9 Classification Using Machine Learning

In machine learning, classification is a type of supervised learning that includes giving labels or categories to data based on their features or properties. The purpose of classification is to create a model that can predict the class of previously unseen or new data based on the patterns acquired from the training data. The data is first divided into training and testing sets in classification. The training set is used to train the model, and the testing set is used to assess the model's performance. The class labels are included in the training data, and the algorithm learns to associate the input features with the proper class. Classification problems are classified into two types: binary classification and multiclass classification. The objective of binary classification is to divide data into two groups, such as yes or no, positive or negative, and so on. The goal of multiclass classification is to divide data into more than two groups, such as different sorts of animals, different types of items, and so on[149].

Classification methods include decision trees, logistic regression, support vector machines (SVMs), k-nearest neighbors (k-NN), and neural networks. Each algorithm has advantages and disadvantages, and the algorithm used is determined by the nature of the data and the specific task at hand. Spam detection, picture classification, medical diagnosis, sentiment analysis, and fraud identification are just a few of the practical applications of classification. It is a valuable tool in data science and machine learning, and its accuracy and performance have improved dramatically in recent years as algorithms and processing capacity have advanced[150].

2.9.1 Support Vector Machines (SVMs)

Support Vector Machines (SVMs) are a powerful class of supervised machine learning algorithms for classification and regression tasks. In classification, SVMs aim to find an optimal hyperplane in a high-dimensional feature space that best separates data points belonging to different classes while maximizing the margin between them. This hyperplane often called the decision boundary, is selected to pass through the support vectors, which are the data points closest to the boundary and are most challenging to classify correctly, as shown in Figure (2.20) [151]. SVM employs a technique known as kernel trick to translate the data into a higher-dimensional space where it can be separated linearly. This is accomplished by using a non-linear function on the original feature space. Linear, polynomial, and radial basis function (RBF) kernels are the most widely utilized. SVM is a sophisticated technique for classification and regression applications that has been used successfully in bioinformatics, text classification, and picture recognition[151][152].

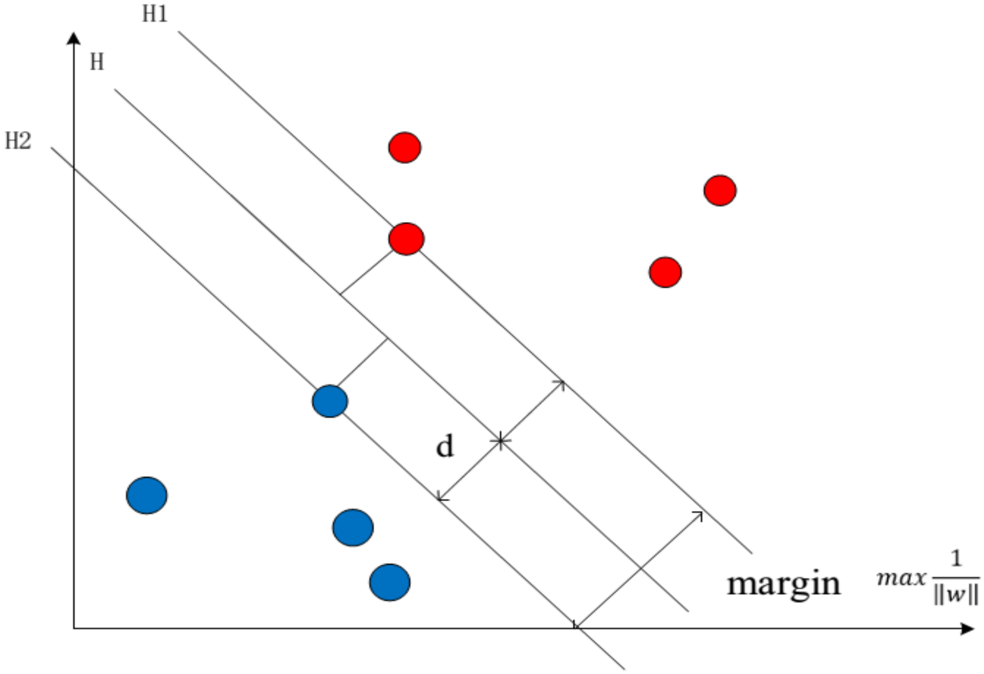


Figure (2.20): Support Vector Machine[153]

2.9.2 Logistic Modeling

A statistical technique called logistic modeling is used to model and evaluate data that has binary outcomes. In this case, the response variable has two levels: success or failure, yes or no, 0 or 1, etc. To comprehend the relationship between a set of independent variables and the likelihood of the occurrence of the binary outcome, it is frequently employed in a variety of sectors, including marketing, economics, healthcare, and social sciences. The logistic model is based on the logistic function, which is a sigmoidal curve that can take any value between 0 and 1. The logistic function is defined as Equation (2.17) [67][28].

$$f(x) = \frac{1}{1 + e^{-\times}} \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots..\dots (2.17)$$

where × is a linear combination of predictor variables and the coefficients, also known as the log-odds or logits. The log-odds can be expressed as Equation (2.18).

$$logit(p) = \log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \beta_1 x_1 + \cdots + \beta_p x_p \dots\dots\dots (2.18)$$

where p is the probability of the binary outcome, x_i are the predictor variables, and β_i are the coefficients that represent the change in log-odds for a unit change in the corresponding predictor variable, keeping all other predictors constant. It uses a method called "maximum likelihood estimation" to figure out the coefficients of the logistic model. This method tries to find the values of the coefficients that make the most sense given the model and the data. In the logistic model, this means figuring out the likelihood function, which is the product of the probabilities of the events that were seen[154].

2.10 Evaluation Measures

Evaluation measures are metrics or methods that are used to measure how well a system, model, or algorithm works by comparing its inputs and results to the real world or a benchmark. In machine learning, evaluation methods are used to judge how well the model made its predictions. Any system requires the evaluation of any model. When one metric is used to evaluate a model, it may produce a solid skill score and satisfying tolerable results, but when other metrics are used, the results are terrible. Most of the time, classification accuracy is employed to assess a model's performance, although this metric does not represent a model's genuine judgment. As a result, more than one evaluation metric must be used[155][156].

2.10.1 Measures of Performance

Evaluating a deep learning or machine learning model's performance is one of the main steps in building an effective model. It is not possible to accurately assess the model's performance by using just one evaluation metric. As it was already indicated, the model can perform admirably with one measure, such as "accuracy score," but poorly with another, such as "logarithmic loss." Accuracy, Logarithmic Loss, Confusion Matrix, Area under Curve, F1 Score, Mean Absolute Error, and Mean Squared Error are a few examples of the various evaluation metrics that are accessible [85][157].

A) Accuracy Measure

Accuracy is a good measure if the dataset is balanced (this means that the number of positive and negative examples is equal or close). Accuracy is calculated by dividing the total number of forecasts by the number of right predictions, as shown in Equation (2.19).

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made} \ldots \ldots . (2.19\ )$$

If the training set has the same number of each class, this measure gives a good rating. That means that classification accuracy works well, but it doesn't make sense to get a high level of accuracy[157].

B) Logarithmic Loss

In order to penalize incorrect classifications, a measurement that known as logarithmic loss (log Loss) assigns a probability to each class for all samples. With multi-class classification, this metric performs well. The Log Loss' range-bound is [0, ∞], and the accuracy is higher when the value is closest to 0 and lower when it is further from 0. As a result, the classifier's accuracy can be increased by reducing Log Loss. According to Equation (2. 20), the Log Loss for N samples belonging to M classes is determined[157][85].

$$LogarithmicLoss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} * log(p_{ij}) \ldots\ldots\ldots\ldots.. \quad (2.20)$$

Where N is the number of samples.

M is the number of classes.

yij: specifies whether or not sample i belongs to class j.

pij: denotes the likelihood that sample i belongs to class j.

C) Confusion Matrix CM

The confusion matrix is one of the most essential approaches for describing the model's overall performance. Table (2.2) represents the CM of a binary classification model with samples belonging to the two classes "YES or NO", such that: when the model predicted YES and the actual output was also YES, this is called True Positive (TP), when the model predicted NO and the actual output was NO, this is called False Negative (FN), when the model predicted YES and the actual output was NO, this is called False Positive (FP), and finally when the model predicted NO and the actual output was YES, this is called True Negative (TN)[157][85].

Table (2.2): The Confusion Matrix [157]

		Predictive model	
		Yes	No
Actual class	Yes	**True Positive** (TP)	**True Negative** **(TN)**
	No	**False Positive** (FP)	**False Negative** (FN)

The matrix accuracy is calculated by averaging the main diagonal values using the Equation (2.21) [157][103].

$$Accuracy = \frac{TruePositives + \ FalseNegatives}{Total\ Number\ of\ Samples} \ldots\ldots\ldots\ldots(2.21)$$

D) Area Under Curve (AUC)

AUC is the region below the False Positive Rate vs. True Positive Rate plot curve at various points in the interval [0, 1] that represents the probability that the classifier model will grade a randomly selected positive example higher than a randomly selected negative example. With an increased value of AUC, the model's performance is enhanced. This measure is one of the most commonly employed metrics for evaluating binary classification problems [157] [158].

- **True Positive Rate (Sensitivity):** It is the average of all correctly positive data points and all positive data points, and the result is in the range [0,1], as shown in Equation (2.22).

$$TruePositiveRate = \frac{TruePositive}{FalseNegative + TruePositive} \ldots (2.22)$$

- **False Positive Rate (Specificity):** It is the average of all incorrectly negative data points, and the resulting value is in the range [0,1], as shown in Equation (2.23).

$$FalsePositiveRate = \frac{FalsePositive}{FalsePositive + TrueNegative} \ldots . (2.23)$$

E) F1 Score

The F1 Score reflects the Harmonic mean of precision and recall measures to assess a test's accuracy, which determines how many times it correctly classifies and is resilient by not ignoring a large number of instances of the model. F1 Score has [0, 1] ranges, with higher values indicating better performance. The F1 Score can be stated analytically as shown in Equation (2.24), the precision as shown in Equation (2.25), and the recall as shown in Equation (2.26) [157][159].

$$F1 = 2 * \frac{1}{\frac{1}{Precision} + \frac{1}{Recall}} \quad \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \; (2.24)$$

- **Precision:**

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives} \quad \ldots (2.25)$$

- **Recall:**

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} \quad \ldots . (2.26)$$

F) Mean Absolute Error (MAE):

The average of the difference between the actual and projected values is the mean absolute error[160]. It calculates the distance between the predictions and the actual output, but it does not identify the direction of inaccuracy under or over the predictive data [157][100]. MAE is mathematically represented in Equation (2.27):

$$MeanAbsoluteError = \frac{1}{N}\sum_{j=1}^{N}\left|y_j - \hat{y}_j\right| \quad \ldots\ldots\ldots\ldots\ldots \quad (2.27)$$

N represents the total number of data points, y_j represents the actual value, $\hat{y}_j$ represents the predicted value.

G) Mean Squared Error (MSE):

The mean squared error (MSE) is the square of the difference between the actual and expected values. The basic goal of MSE is to make it easier to compute the gradient, which allows the influence of greater errors to be more powerful than a smaller error based on taking the square, which enables the model to focus on the larger errors [157][161]. MSE is theoretically represented as shown in Equation (2.28):

$$Mean\ Squared\ Error = \frac{1}{N}\sum_{j=1}^{N}\left(y_j - \hat{y}_j\right)^2 \quad \ldots\ldots\ldots\ldots\ldots\ldots (2.28)$$

2.10.2 K-Cross Validation

Cross-validation is a method for estimating the quality of an Artificial Neural Network model by choosing the best set of parameter values (such as the number of hidden nodes, back-propagation learning rate, and so on) and applying them to numerous neural networks. Unreliable Deep Learning model evaluation may occur due to an unjust partition of datasets into testing and training sets that are not representative of all data. The performance metric is reported by k-fold cross-validation, which then computes the average of the values in the loop[162]. This method involves repeating the entire procedure (training and testing) several times with different data samples based on the K value. This strategy is computationally expensive, but it does not waste a lot of data, especially when the number of samples is modest. The prediction model is repeated K times in K-fold cross-validation, with 1-K of data picked for testing and the remainder used for training in each iteration. The overall error rate is then calculated by taking the average error of all iterations[162][163]. Figure (2.21) depicts the 5-fold cross-validation approach including training and testing.

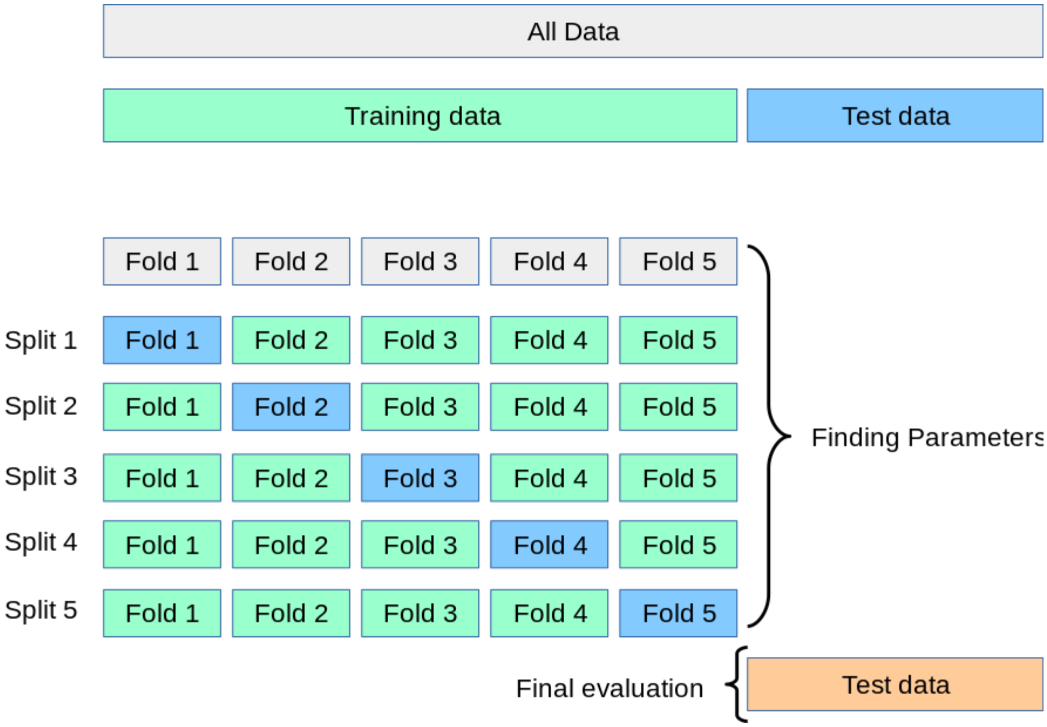


Figure (2.21): 5-Fold Cross-Validation Procedure[164]

Chapter Three

The Proposed System Design

CHAPTER THREE
THE PROPOSED SYSTEM

3.1 Introduction

In this chapter, the operational phases of the proposed system are described in a series of stages. The method extracts features from the videos (dataset). The features are the action unit and face pose, which are unique and vary from person to another. These features are extracted using JAA-Net and hybrid face pose networks, and then a profile is created for a person. A person's profile has all the features through which real and fake videos can be predicted. A prediction process is done using BiLSTTM, where the models are trained on the features that have been extracted.

3.2 Methodology

The proposed system consists of two phases of the deep fake video detection process, as shown in Figure (3.1). The first phase is the capture frame from the dataset, which contains real and fake videos. Videos are pre-processed by taking one frame out of every twenty-four frames and resizing the extracted frames. Then the action unit features are extracted through the JAA-Net model. Face pose features are extracted through the hybrid face pose network. Those features extracted from the first phase are stored in a profile. The profile that has been created contains the action unit and the movement of the head (face pose), which is considered the behavior of the person. The second phase is the prediction, which depends on the accuracy of the features extracted from the first phase. The prediction phase uses BiLSTM, as it is trained on the extracted profile.

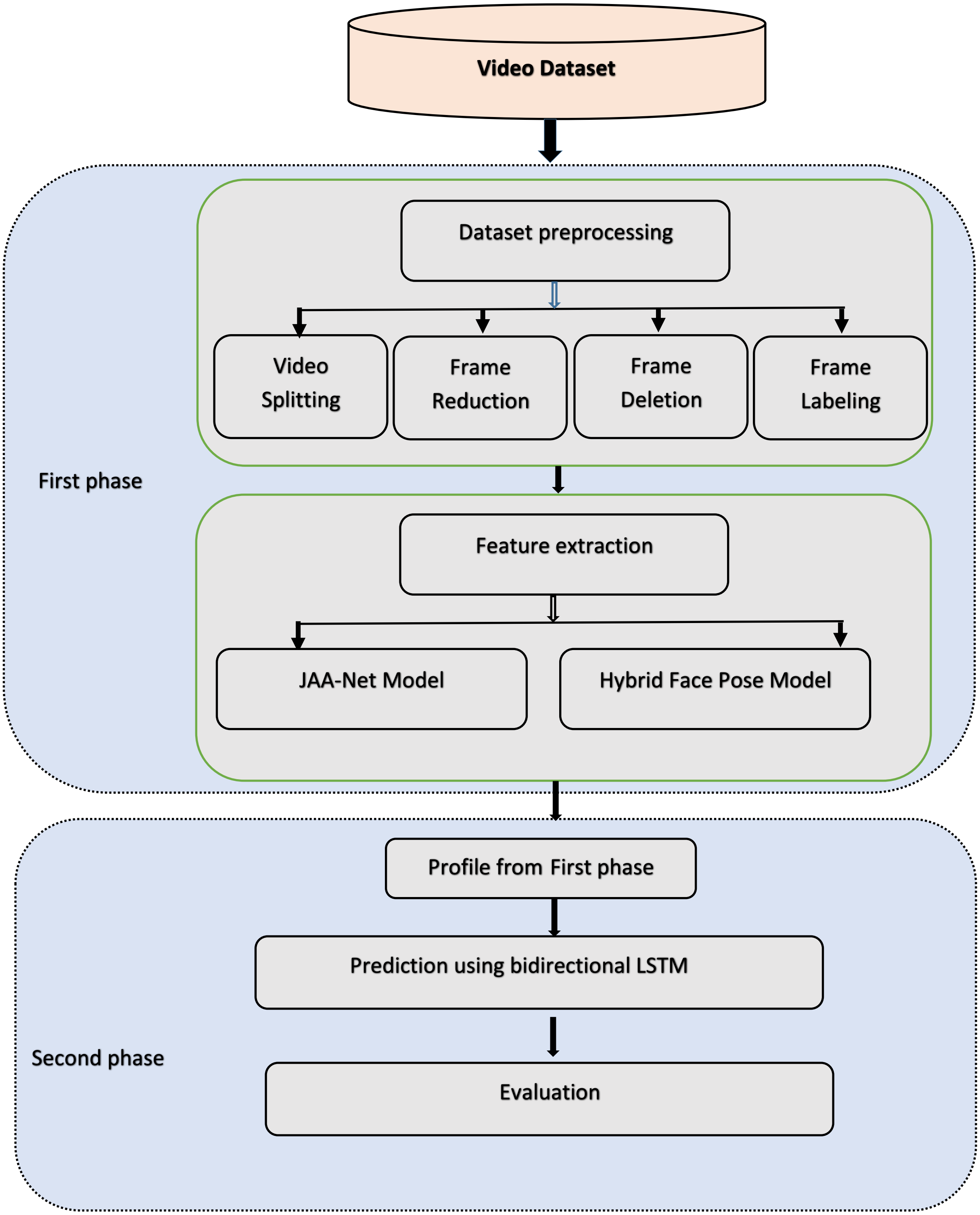


Figure (3.1): Block Diagram of the Proposed System

3.2.1 Dataset

The Barack Obama dataset was selected as the preferred one. The Barack Obama dataset contains videos that are considered efficient and very close to reality, as shown in Figure (3.2). Also, the videos in this dataset haven’t contained artifacts like the rest of the database, such as forensics++, Celeb-DF, deepfake_database, and Deepfake Detection Challenge Actors, as shown in Figure (3.3). These videos are accessible to the public, have good pixel ratios, and are also extremely simple to test for Deepfakes. Therefore, download online videos manually to generate a specialized dataset. The system requires two types of videos to evaluate the proposed hypothesis: the original and the synthetic. To ensure the integrity of the videos, the originals were obtained from the Miller Center's official website. This website broadcasts speeches by U.S. presidents. The size of the dataset used is 227.97 minutes, divided into two groups: real and fake videos.

Figure (3.2): Five example frames of a 10-second clip, (a) the original, (b) synthesis deep fake, (c) comedic impersonator, (d) face-swap deep fake, (e) puppet-master deep fake

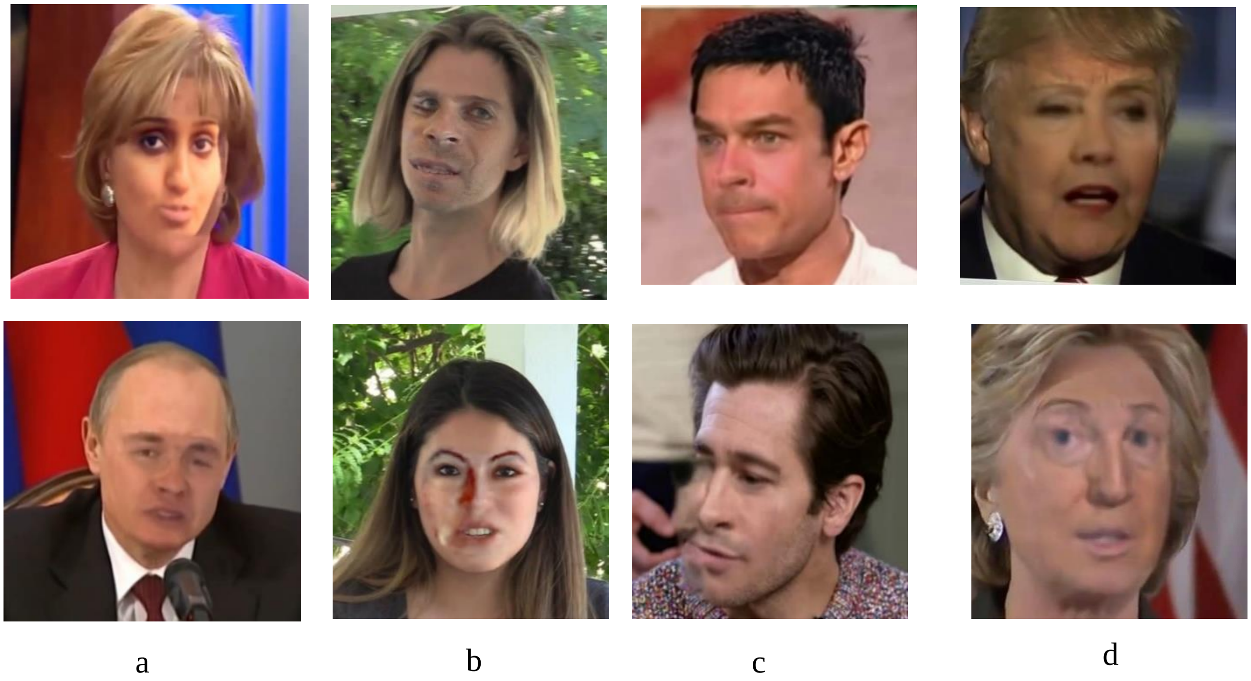


Figure (3.3) Some Samples Demonstrate Accuracy and Efficiency in Database Types, (a) forensics++,(b) Deepfake Detection Challenge Actors,(c) Celeb-DF, and (d) deepfake database

3.2.2 Dataset Preprocessing

Video pre-processing is a necessary procedure in building a prediction model because the dataset of Barack Obama is not in the optimal format to be fed into these models. The preprocessing in this system has been accomplished in four basic steps: (1) video splitting; (2) frame reduction; (3) frame deletion; (4) frame labeling.

3.2.2.1 Video Splitting

The nature of the videos in the dataset for Barack Obama is that they are in MP4 format. The width of the frames is 1280, the height of the frames is 720, and the frame rate is 30 frames per second. The video files are made up of several frames (images). In this step, the video dataset is divided into frames.

3.2.2.2 Frame Reduction

Pre-processing is applied to the videos to get the frames where one in twenty-four frames is read. The important role of this stage is to reduce the computational complexity as well as find features that are distinct from the rest of the features in the videos. The other reason is that the features are very close when all the frames in the videos are read.

3.2.2.3 Frame Deletion

After the process of splitting the video into frames, some frames do not contain the face of Barack Obama. Sometimes a small part of the face appears, and these frames cannot be used to extract features. Frames that haven’t contained features related to expressions and facial poses are eliminated.

3.2.2.4 Frame Labeling

The dataset being worked on contains both real and fake videos. In the process of reading, frames need to be labeled. The vector is extracted from the features in each frame by means of the JAA-Net and hybrid face pose networks. Therefore, when reading real videos, the vectors extracted from the frames must be given a real label. And when reading fake videos, they must be labeled with vectors extracted from the frames and given a fake label.

3.2.3 Feature Extraction

The extraction of features is an important stage in this proposed system because it analyzes the person's behavior. The person's behavior is to find facial expressions, which depend on the action unit and head pose. Models for extracting features are explained in the following sections:

3.2.3.1 JAA-Net Model

JAA-Net was used to extract the action unit for a face from each frame. Some action units are shared in certain proportions between people in normal situations. In the proposed system, twelve action units were relied upon. These action units are considered the most distinctive action units in facial expressions in terms of sadness, anger, surprise, laughter, and others, as shown in Table (3.1).

JAA-Net consists of four distinct modules: hierarchical and multi-scale region learning, face alignment, global feature learning, and local feature learning. More details of the general structure of the JAA-Net model (are in Section 2.8.1).

a- Hierarchical and Multi-scale Region Learning

The first module is the hierarchical and multi-domain learning of the area consisting of four convolutional layers. One convolutional layer is followed by three hierarchical convolutional layers with varying weight-sharing region widths. Specifically, the 8×8, 4×4, and 2×2 patches of the second, third, and fourth convolutional layers are the result of convolution on the equivalent patches of the previous layer. It extracts hierarchical and multi-scale features with the same number of channels as the first convolutional layer by concatenating the outputs of the second, third, and fourth convolutional layers.

Table (3.1): The Types of the Action Units

Action unit	Facial Action Coding System
AU01	Inner Brow Raiser
AU02	Outer Brow Raiser
AU04	Brow Lowered
AU06	Cheek Raiser
AU07	Lid Tightener
AU10	Upper Lip Raiser
AU12	Lip Corner Puller
AU14	Dimpler
AU15	Lip Corner Depressor
AU17	Chin Raiser
AU23	Lip Tightener
AU24	Lip Pressor

b- Face Alignment

The face alignment module consists of three convolutional layers, each of which connects to a max-pooling layer. The output of this module is fed into a landmark prediction network consisting of two fully connected layers of dimension d and N_{align}, where N_{align} is the number of face landmarks. It defines the loss of face alignment as Equation (2.15). The Landmark number used in the proposed system is 49. In this model, the face alignment will be extracted from which the general shape information and the local Landmark information will be extracted. These features will be used to find the action unit location with subsequent modules.

c- Global Feature Learning

The global feature learning module is used to get information about the global structure and texture of the face. It is built the same way as the face alignment module. Both the global output feature and the face alignment feature are used for the final AU detection. These features can add useful information to the local AU features.

d- Local Feature Learning

The local feature learning module consists of two steps: refinement of AU attention and AU local feature learning. The first step is to refine the predefined attention map for each AU with a separate branch, and the second step is to learn and extract the local AU features. Initialization and refined attention map results are the inputs and outputs of the AU attention refinement process. Each AU possesses an attention map corresponding to the entire face with dimensions height/4×weight/4, in which the attention distributions of both predefined ROI and residual areas are improved. Due to symmetry in the face, the designated ROI of each AU has two AU centers, each of which is the center of a subregion. For instance, predicted face landmarks determine the positions of AU centers. If the k-th point of the attention map falls within a subregion of the designated ROI, the attention weight for the i-th AU is initialized as Equation (2.16), this equation shows that the attention weights are going down when the ROI points to move away from the center of the AU. In Equation (2.16), the maximization step is to make sure that v_{ik} is between 0 and 1. If a point is in the middle of two subregions, it is given the highest value of all the initial attention weights that go with it. Note that when $\xi = 0$, the points in the subregions have an attention weight of 1. Any point outside of the subregions starts out with an attention weight of 0.

The JAA-Net network was pre-trained on Dataset Bp4D as shown in 2.8.1. At this stage, JAA-Net is used to detect AU for a dataset of Barack Obama. The process of finding the AU goes using JAA-Net, as shown in Figure (3.4).

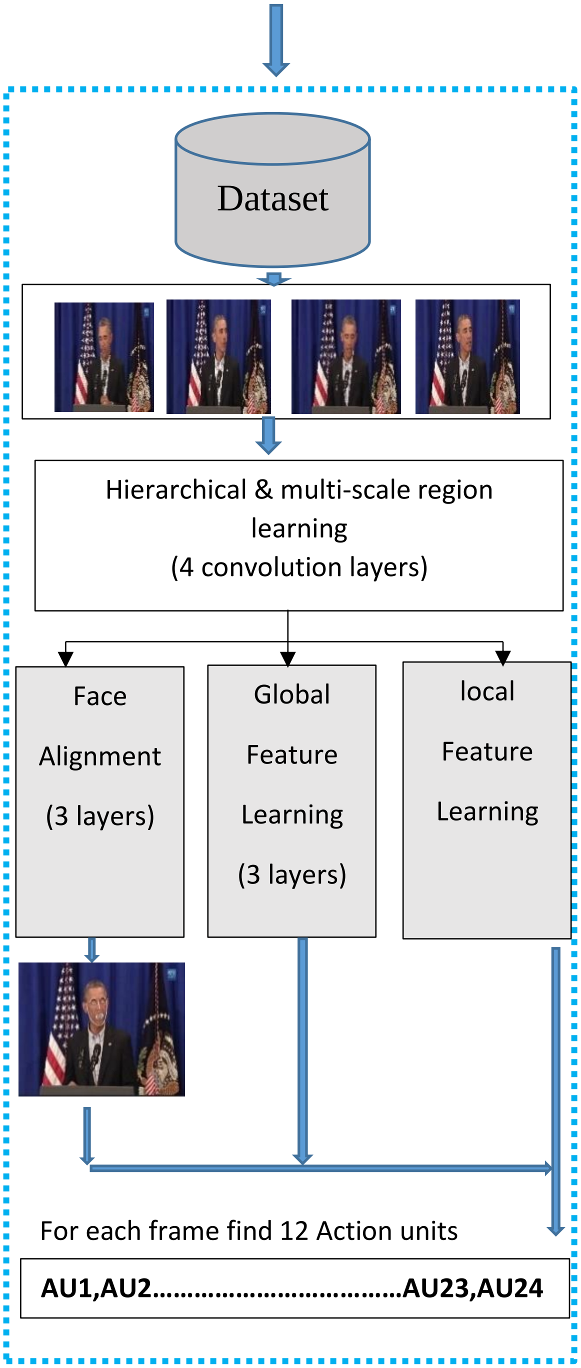

Dataset
Hierarchical & multi-scale region learning (4 convolution layers)
Face Alignment (3 layers)
Global Feature Learning (3 layers)
local Feature Learning
For each frame find 12 Action units
AU1,AU2……………………………AU23,AU24

Figure (3.4): Block Diagram of Action Units' Detection

Video files are being entered into the JAA-Net network. The model converts the videos into frames and then inserts them into a network. The network finds twelve action units for each frame it reads. The value of each action unit ranges between 0 and 1, which represents the percentage of effectiveness of this action unit on the face. Each action unit in the face moves at a certain rate, depending on facial expressions in cases of sadness, anger, disgust, fear, neutral, etc. The value of each action unit may appear in large proportions in some frame, depending on emotional states. Action unit 24 has a clear value in the case of lip pressure. Action 23 also appears in emotional situations in which the lips are tightened. As for action unit 17, it occurs in some facial expressions in which Chain raiser is present. For example, AU07 One of the AUs listed in anger prototypes is "lid tightener," but it is not only related to anger. Lid tightener can also be used to show that someone is tired, focused, frustrated, having trouble seeing, etc.

In this dissertation, two other methods are used in the feature extraction process, which are (Logistics and SVM). These two methods hadn't given the desired results and are used in comparisons in the fourth chapter. Firstly **,**the logistic model, also referred to as the logit model, is a statistical model that calculates the event's log-odds as a linear combination of one or more independent variables. It assesses the likelihood that an event will occur (for more details, see section 2.9.2). The logistic method relies on a series of steps to define the action unit (face detector, facial landmark detectors). Logistic modeling defines the twenty action units. This differs with the JAA-Net in terms of the number of types of action unit**.** Secondly, SVM modeling also depends on a series of steps that define the action unit (for more details, see section 2.9.1). The same steps are used for the logistics model also twenty action unit. The same step is taken from the face detector and facial landmark detectors and finding Histograms of Oriented Gradients (HOGs). As for the method of classification between the features extracted from the frames to find the action unit based on the DISFA Plus dataset, it is a support vector machine. Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression, and outliers' detection. Type SVM that using in this dissertation linear Support Vector Classification.

3.2.3.2 Hybrid Face Pose

Face pose is one of the most reliable features for analyzing a person's behavior and can identify fake and real videos. A method that based on Six Degrees of Freedom(6DOF) has been proposed to find out the features of the facial pose. This method guesses the 3D position of the face without using the landmark. The 6DOF has more information than using a bounding box label for a face. The 6DOF deals with six dimensions in finding features (Yaw/Pitch/Roll) as shown in Figure (3.5).

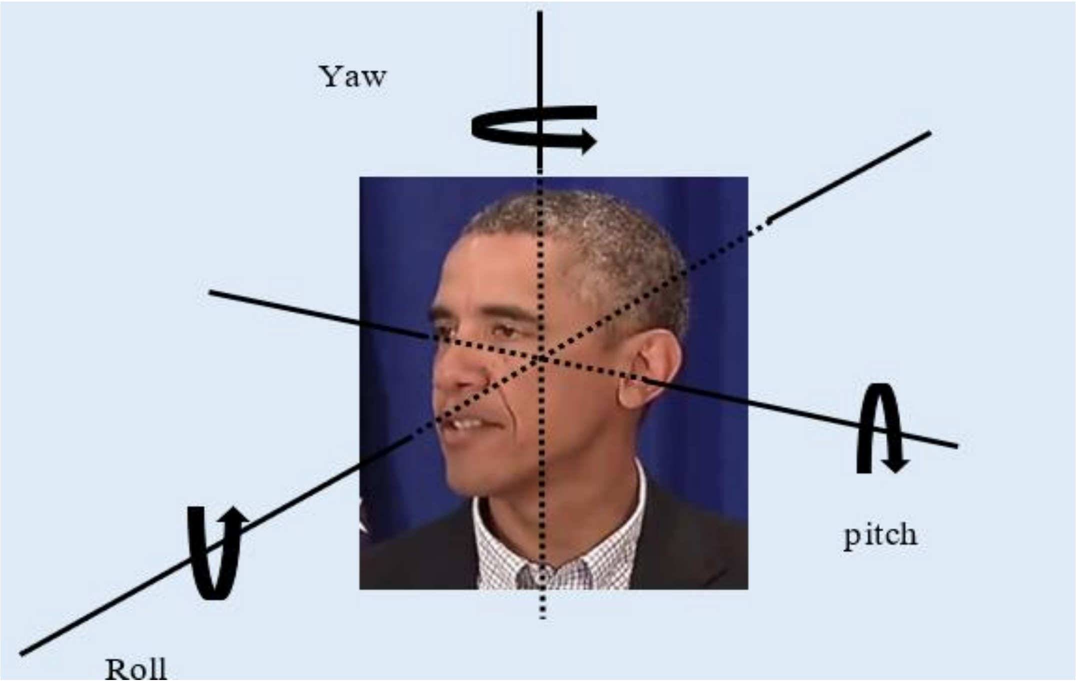


Figure (3.5): Face Pose Features

Estimating 6DoF pose is a 6D regression problem, which is clearly smaller than even 5-point landmark detection (5×2D landmarks is 10D), let alone standard 68 landmark detection (68 ×2D landmarks is 136D). Finding the features of the position of the face is done by projecting a 3D face shape onto a 2D face without using the landmark, where the 6DoF of the face is estimated, and this method is the easiest for facial landmark detection. A 3D face shape projection of the face in the frame is based on the ground truth through which the features of the face pose are found, as shown in Figure (3.6). In the proposed system, a hybrid model is relied on to find its face pose as shown in the algorithm (3.1).

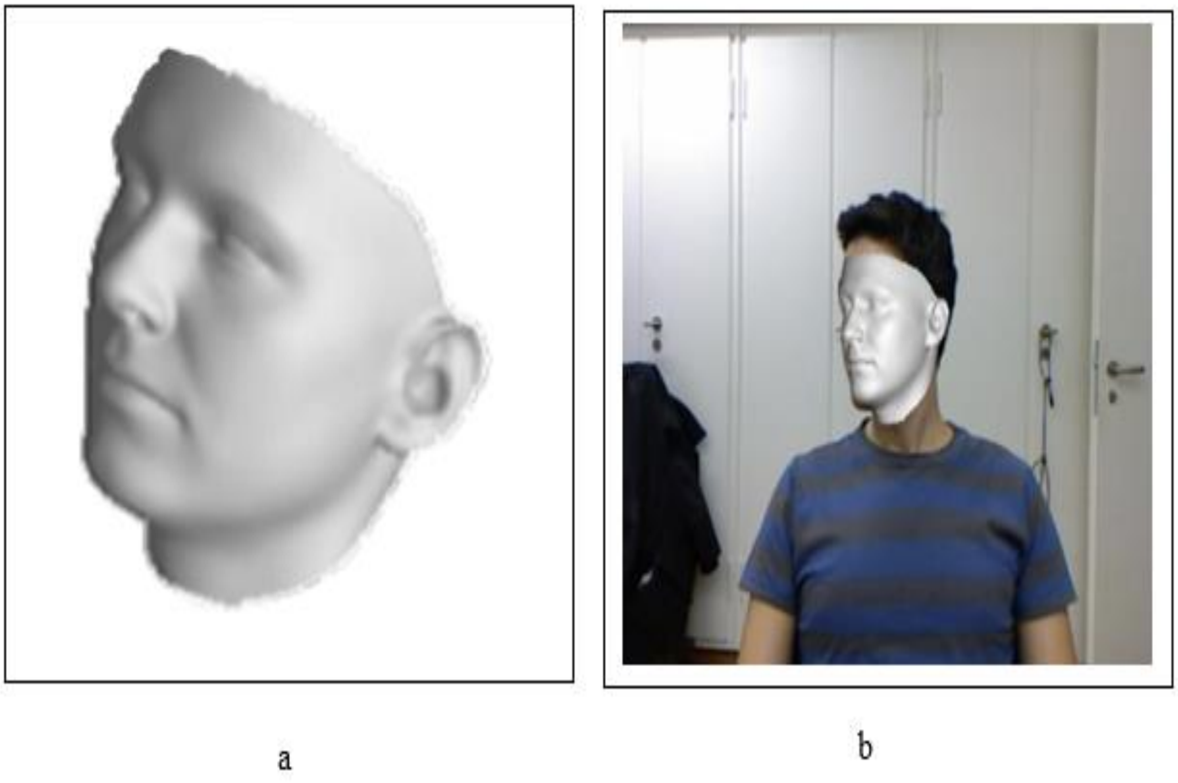


Figure (3.6): The 3D Face and The Projection, (a)3D Face Pose , (b) A 3D Face Shape Projection in 2D Face

Algorithm (3.1): Steps of hybrid face pose

```
Algorithm name: Hybrid face pose.
Input:  frames(images).
Output: face pose. // represent of features (Yaw/Pitch/Roll)
begin
1. Apply the feature pyramid network (FPN) network for feature extraction.
    - Apply the bottom-up pathway to compute a feature hierarchy using the
      Resnet18 network.
    - Apply the top-down pathway for the high-level features.
    - Apply lateral connections for connections between the bottom-up and
      top- down pathways using a 1 × 1 convolution filter to reduce channel
      depth to 256-d.
2. Using Region Proposal Network (RPN) based on Faster R-CNN.
   // This step is face   detector, in order to anticipate whether each region has a face
      and its boundary box
3. Extract the feature from each   RPN with the region of interest (ROI) pooling.
   // This step is used to find the final region of interest from which the final
      classification will be made.
4. Classification of the feature extracted from each ROI pooling and then sending
   them to a normal face/non-face (faceness) classifier, a 6DoF face pose regressor,
   and two separate heads
5. The face pose regressor and the face/non-face classifier head are used to find
   features (yaw, pitch, or roll).

End algorithm
```

The general structure and main steps of the algorithm will be explained. The 6DoF is estimated for each face (i) identified in the frame (image)(I). the method uses $h_i \in R^6$ which indicates the pose of the face:

$$h_i = (r_x, r_y, r_z, t_x, t_y, t_z)$$

Where (r_x, r_y, r_z) is a rotation vector and (t_x, t_y, t_z) represent the translation of the 3D face.

The second step is to project a 3D face shape onto the two-dimensional face in the frame. This step is important in calculating its facial pose based on existing labels in a dataset. From these labels, figures are calculated for face pose. The Equation (3.1) model determines the procedure for calculating the projection points of a three-dimensional shape on a two-dimensional face.

$$[Q, 1]^T \simeq K[R, t][P, 1]^T \qquad \dots\dots\dots\dots\dots\dots\ (3.1)$$

This equation represents the standard pinhole camera model where K is the intrinsic matrix. The intrinsic matrix converts camera coordinates from three dimensions to a uniform two-dimensional image space. R and t represent the three-dimensional rotation matrix and the translation vector, respectively, obtained from h. A matrix that represents n 3D points on the surface of the 3D face form is denoted by the notation $P \in R^{3\times n}$. The matrix representation of 2D points that have been projected from 3D onto the image is denoted by $Q \in R^{2\times n}$. Importantly, it's easy to get a face detection bounding box by just picking the bounding box that has the projected 2D points, Q. Notably, this method allows for more precise bounding box tightness and morphology management. In particular, the pose connects a 3D object with known geometry to a face region in the image, so it can select to adjust the face bounding box sizes and shapes such as including more of the forehead by extending the box in the correct direction, an invariant of the pose.

The algorithm consists of a number of steps:

a- feature pyramid network (FPN) using Resnet 18 network

The first step is feature pyramid network (FPN) using Resnet 18. The plan is to use ConvNet's pyramidal feature hierarchy, which has semantics at multiple levels, from the lowest to the highest, to build a feature pyramid with high-level semantics built into it. This step makes a possible face area with a clear pyramid that shows where possible faces might be in the image. The Resnet 18 network consists of five convolutional layers, which is considered the backbone of the FPN network as shown in Figure (3.7).

The bottom-up pathway is the backbone of ConvNet's feedforward computation, which computes a feature hierarchy consisting of many scaled feature maps with a scaling step of two. Several layers within the same network stage typically generate output maps of the same size. It is pyramidal, with each stage defining one pyramid level. It uses the final layer output from each stage as a reference set of feature maps to enrich the pyramid's construction. The model takes advantage of the feature activations generated by the last residual block in each stage. The strides of these residual blocks are noted to be 4, 8, 16, and 32 pixels relative to the original image, and their output is labeled as "C2, C3, C4, C5" for conv2, conv3 conv4, and conv5. Due to its high memory requirements, Conv1 is left out of the pyramid.

Lateral connection and a top-down pathway: The top-down pathway up samples spatially coarser but semantically stronger feature maps from higher pyramid levels to conceive things with higher resolution features.

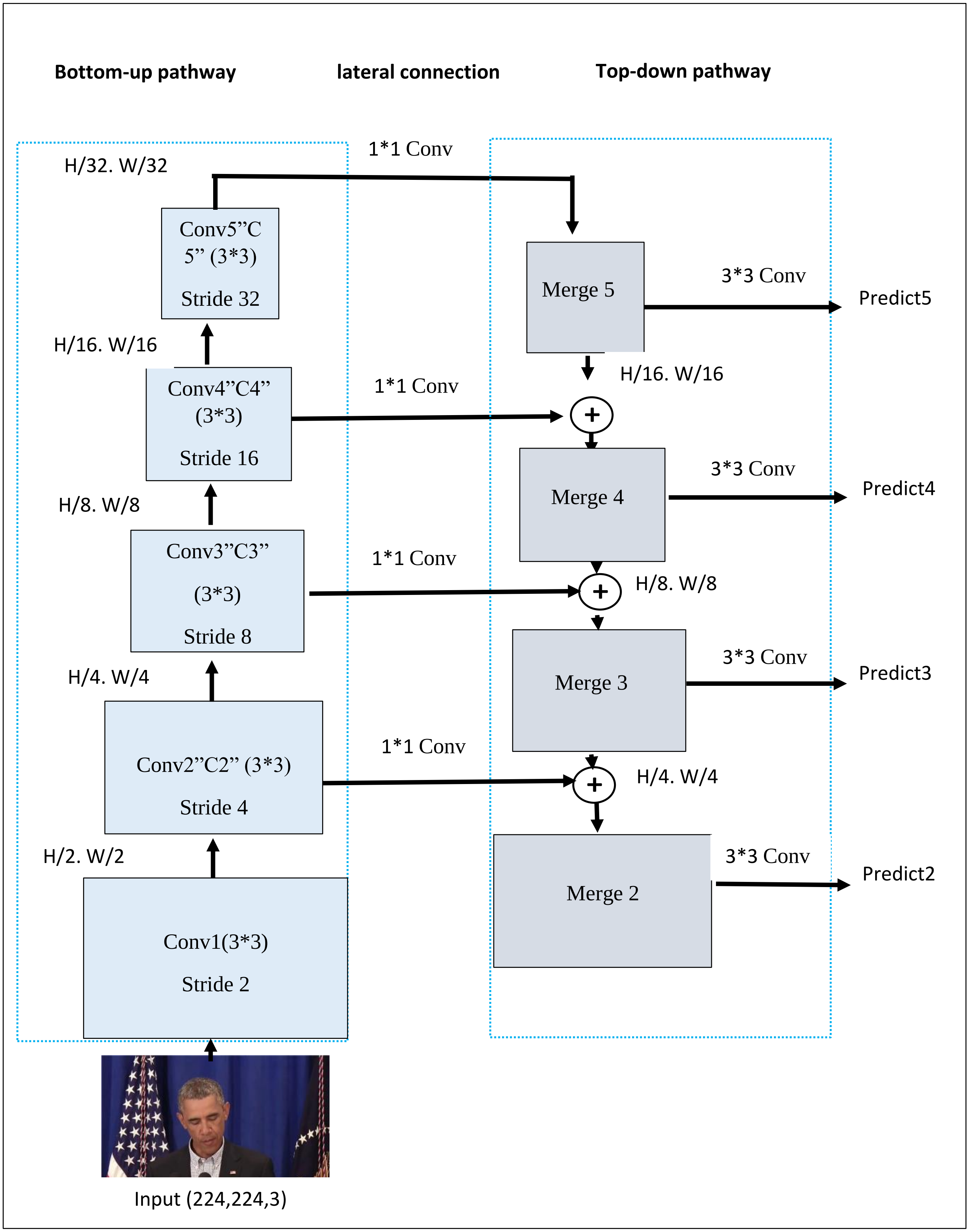


Figure (3.7): Architecture Feature Pyramid Network using Resnet 18

By lateral connections, these features are then improved with features from the bottom-up pathway. Each lateral connection combines feature maps from the top-down and bottom-up pathways that are the same spatial size. The bottom-up feature map has lower-level semantics, but because it is subsampled less frequently, its activations are more precisely localized.

The top-down pathway uses a 1x1 convolutional filter to create the feature maps Merge5, Merge4, Merge3, and Merge2 by reducing the number of channels in the feature maps from the higher-level layers (C5, C4, C3, C2) to 256 dimensions. This step is crucial for achieving feature pyramids that can effectively handle objects at various scales. It helps in fusing high-level semantic information with lower-level details. It uses the nearest neighbor up-sampling to up-sample the preceding layer by two as it proceeds down the top-down path. In the bottom-up pathway, it once again applies a 1×1 convolution to the associated feature maps. Then it adds them element-wise. Its convolutions all merged layers in a 3×3 fashion. When combined with up-sampled layer, this filter lessens the aliasing effect.

b-Region Proposal Network (RPN) based on Faster R-CNN

FPN extracts feature maps, which are then fed into a face detector called RPN. In order to anticipate whether each region has a face and its boundary box, RPN uses a sliding window over the feature maps. The RPN architecture applies a 3×3 convolution filter on feature maps for each scale level, then a 1×1 convolution for faceness predictions and boundary box regression. These layers are the RPN head. Feature map heads apply all scale levels, as shown in Figure (3.8). Using Equation (3.1), it derives projected bounding boxes using the 6DoF ground-truth pose labels.

c- Extract the feature from each RPN with the region of interest (ROI) pooling

In the previous step, each level will produce facial features in the proposed region. This step is used to find the final region of interest from which the final classification will be made. In the final classification, the pose of the face is ideally located, after which the 3D face is projected onto the face that has been determined. After the process of projecting a three-dimensional face onto a two-dimensional face, the degree of its face pose is calculated. To find the best face location from the set of face levels obtained from the previous step, use a small net. The small net consisted of two fully convolution layers, and after each layer, SoftMax is used. This feature is extracted from each ROI pooling and then to be send them to a normal face/non-face (faceness) classifier, a 6DoF face pose regressor,and two separate heads.

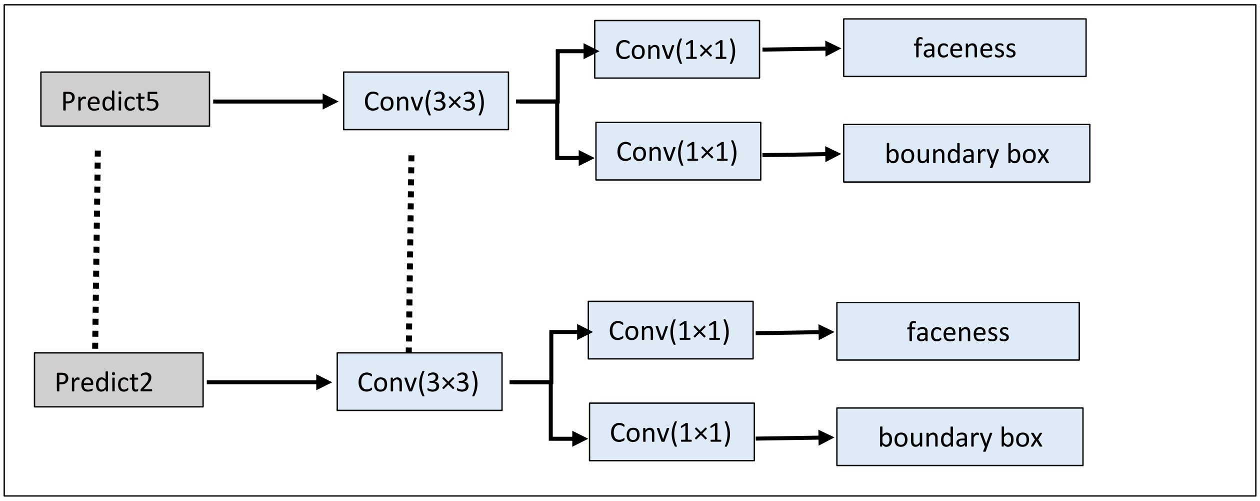


Figure (3.8): Architecture of Region Proposal Network

In Training Losses, the face pose regressor and the face/non-face classifier head are trained simultaneously. The model uses the following multi-task loss for each proposal, as shown in Equation (3.2).

$$loss = loss_{cls}(p_i, p_i^*) + p_i^* \,.\, loss_{pose}\left(h_i^{prop}, h_i^{prop*}\right) + p_i^* \,.\, loss_{calib}\left(Q_i^C, Q_i^{C*}\right) \qquad \ldots\ldots\ldots \; (3.2)$$

$loss$ which includes these three components:

Face a loss of classification ($loss_{cls}$) which uses the common binary cross entropy loss, or $loss_{cls}$, to categorize each proposal, where p_i is the likelihood that proposal i has a face and p_i^* is the ground-truth binary label for that proposal (1 for face and 0 for background). The intersection over union (IoU) between each proposal and the ground-truth projected bounding boxes is used to calculate these labels. $loss_{cls}$ is the sole loss it use for negative proposals that have no faces (p_i^* = 0). Positive that have faces, (p_i^* = 1).

Face pose loss ($loss_{pose}$) compares ground truth with a 6DoF face pose estimate. Where h_i^{prop} is the predicted face pose for proposal i, h_i^{prop*} is the ground-truth face pose.

Calibration point loss ($loss_{calib}$) measures the projection of a 3D shape onto a 2D face. It compares between ground truth pose and predicted face pose. Where Q_i^C projected 3D using predicted pose h_i^{prop}, Q_i^{C*} projected using the ground-truth pose h_i^{prop*}.

The model goes through two stages, namely the pre-training stage and the feature extraction stage. In the pre-training stage, the 300W-LP dataset is used. 300W-LP is a dataset including 122,450 photos of synthesized head positions from 300W. Training pose rotation labels are generated by converting the 300W-LP ground truth Euler angles to rotation vectors, and pose translation labels are generated by standard means using the ground truth landmarks. The second stage is to extract the face pose feature from the dataset and find its behavior. Video files are being entered into the Hybrid face pose network. The model converts the videos into frames and then inserts them into a network. The network finds features (Yaw/Pitch/Roll) for each frame read. The value of each features ranges between (−90, +90), which represents the percentage of effectiveness of these features on the face. The value of each face pose may appear in large proportions in some frames, depending on head movement.

3.2.4 Prediction using Bidirectional LSTM

The second phase of the proposed system is prediction. This phase comes after extracting the action unit and face pose of the POI and creating a profile for it. The features in the profile are imbalanced because the number of fake videos is less than the number of real videos, which is because some real videos have been added to the dataset. These added videos contain the largest number of action units and facial poses. This set of imbalanced data is processed using SMOTE (Synthetic Minority Oversampling Technique). The action unit and face pose are used to train the deep learning artificial neural network. The videos are evaluated based on the precision of the BiLSTM results, which are derived from a profile containing features that represent Barack Obama's distinctive behavior as shown in the algorithm (3.2). In the proposed system, many prediction models are used, such as (ANN, SVM, LSTM, and GRU) and the best prediction network was BiLSTM.

Two long-short-term memory (LSTM) models with opposing directions make up a BiLSTM. In addition to training from input to output, models are also trained from output to input. Each unit in BiLSTM is split into two identical units that share an output and an input. As a result, BiLSTM offers a number of benefits while learning extensive behavior features. The reason behind relying on the BiLSTM neural network in the prediction process is that it has the ability to predict the behavior of the object based on the previous and subsequent features. The BiLSTM Network training process is carried out by taking a vector (x) from each record(R) of the object's profile. Each vector contains 15 features that have been extracted from the previous stages.

Algorithm (3.2): Steps of Prediction

```
Algorithm name: Prediction.
Input:  Profile of features Vectors (action unit and face pose).
             // outputs of the first phase.
Output: Percentage of how many video frames are real or fake and to
           detect real or fake frames.
begin
1. Read a Profile that contains a set of vectors and each vector contains
     Fifteen features.
2. Apply SMOTE function to handle imbalanced instances(real and fake)
     in profile.
3. build the architecture of pridction model
    3.1 make the LSTM in the first layer with the following parameters:
              -takes Fifteen features as input.
             - kernel regularization term with an L1 regularization factor
           of  0.001
   3.2 make the batch normalization  in second layer . It helps normalize
         the  activations of the previous layer.
   3.3 make (opposite LSTM )in the third layer ; that processes the
         input sequences in both forward and backward directions.
   3.4 Adding dropout layer (the probability is 0.3)
   3.5  Adding  batch normalization  layer.
   3.6 Adding another layer is Dense layer; this layer is a fully connected layer.
   3.7 Adding classification layer (one neuron) with sigmoid function.
4. training model by Updating parameters of the model using the binary
    cross-  entropy loss function with the Adam method.
5. Compute evaluation metric (accuracy) for traind model.
6. The result of the Decision
        - The prediction of fake or real video, by each frame that if
        contains facial expressions with behavior similar to that of real
        facial    expressions is determined to be real; otherwise, it is
        determined to be fake.
        - compute the Percentage of how many video frames are real or fake
         and detect real or fake frames.
End algorithm
```

As for the output, it is (y), which contains two values, it is either real or fake, as shown in Figure (3.9). The BiLSTM network stores both earlier information and later information as the current time basis of feature series by processing the feature in forward and reverse directions using two independent hidden layers connected to the same output layer. Thus, compared to uni-direction LSTM, theoretical prediction performance is better.

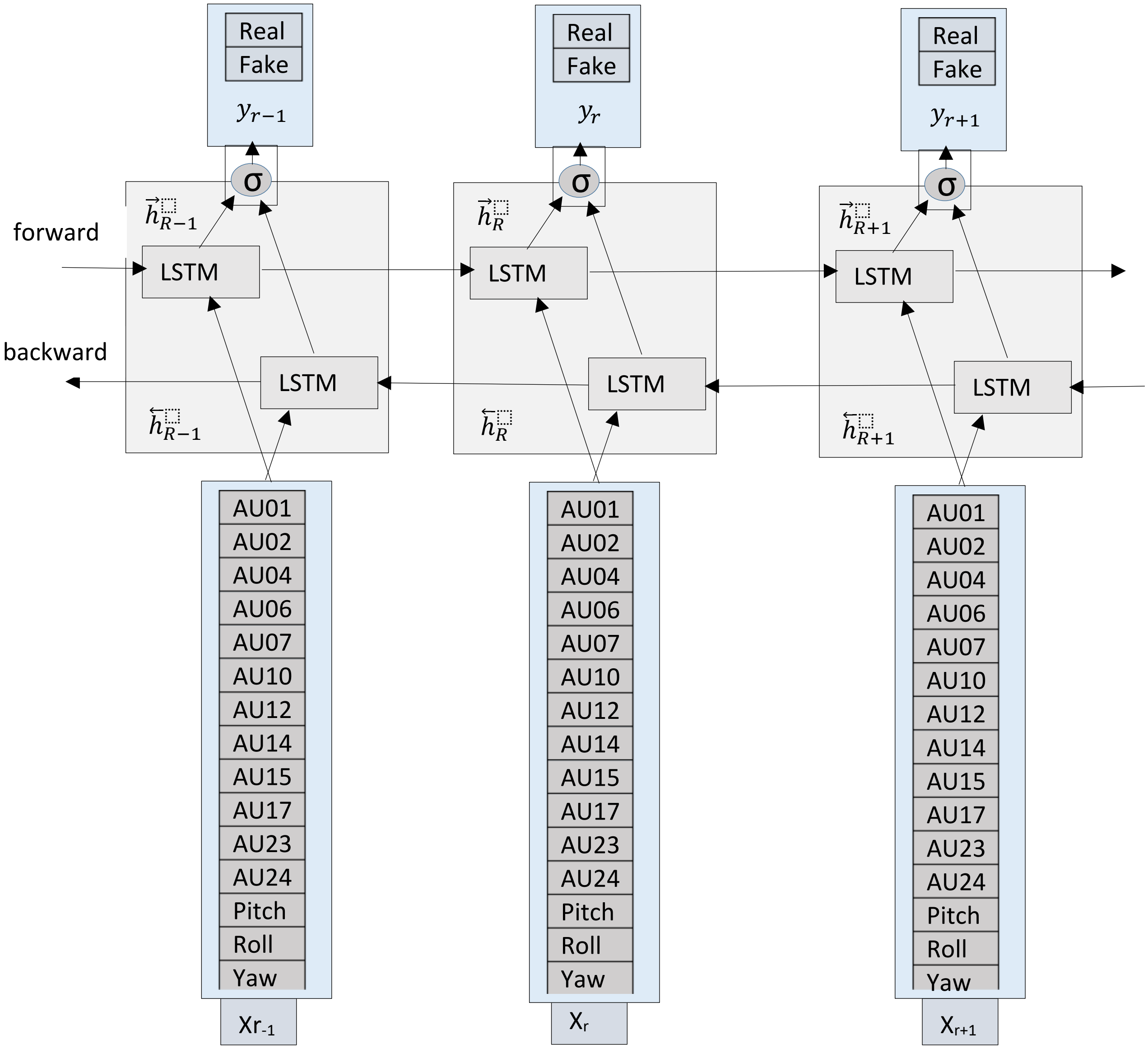


Figure (3.9): Architecture of the Prediction Model using BILSTM Network

The activation output of the forward and backward hidden layers is included in the hidden layer output of BiLSTM. Using the Equation (2.11), (2.12), and (2.13), BiLSTM network has been structured and these equations have a role in predicting a person's behavior. The BiLSTM network is based on the LSTM network, which consists of three gates. For more details, see Section (2.7.3.3).

In the proposed system the network consists of five layers, where the first layer takes fifteen features and the value of the class is one for the LSTM network. It includes a kernel regularization term with an L1 regularization factor of 0.001. In the context of BiLSTM, kernel regularization refers to the application of a penalty term on the weights of the LSTM layer. Regularization techniques are used to prevent overfitting and improve the generalization ability of the model. Specifically, kernel regularization in a BiLSTM involves adding a regularization term to the weights (kernels) of the LSTM layer. The regularization term is multiplied by a regularization factor and added to the loss function during training. This encourages the model to learn simpler and more generalized representations by discouraging large weight values. L1 regularization, also known as Lasso regularization, encourages sparsity in the learned weights by adding the absolute values of the weights to the loss function.

The second layer is the batch normalization layer; this layer is added after the LSTM layer. It helps normalize the activations of the previous layer, improving the stability and speed of training.

The third layer is the BiLSTM layer; this layer is a variant of LSTM that processes the input sequence in both forward and backward directions. It has units and includes dropouts with a rate of 0.3. also has the same input as the first layer.

The fourth layer is the batch normalization layer. This layer is added after the Bidirectional LSTM layer. The last layer is Dense layer; this layer is a fully connected layer. It uses the sigmoid activation function. The model is then compiled with the binary cross-entropy loss function, the Adam optimizer, and accuracy as the evaluation metric.

3.2.5 The Result of Decision

The decision of the proposed system after training the BiLSTM network is to give a percentage of how many video frames are real or fake and to detect real or fake frames. After training the network, the weights are stored, through which the prediction stage takes place. The prediction stage takes videos, converts them into frames, and then extracts facial expression features (action units and facial poses). These features are considered biometric, and any frame that contains facial expressions with behavior similar to that of real facial expressions is determined to be real; otherwise, it is determined to be fake. Therefore, the final result is to detect the real and fake frames, in addition to how many there are in the videos, giving the percentage of the video, how much is real and how much is fake.

Chapter four

THE EXPERIMENTAL RESULTS

CHAPTER FOUR

THE EXPERIMENTAL RESULTS

4.1 Introduction

This chapter shows the results of the experiments and how each step of the proposed system works. It also includes a description of the proposed system 's dataset and the hardware and software needed to make the method work. The results of each stage are arranged based on how they appeared in the previous chapter. The effectiveness of the proposed system is evaluated with different parameter values, and the implementation results are analyzed. It is important to note that evaluating the proposed system's phases is implemented separately. A public dataset is used as a case study to determine this model's behavior. In the following sections, the datasets are utilized by the proposed system, and the method requirements are described. Other sections clarify the results obtained at each stage of the proposed system.

4.2 System Requirements

High computer resources are needed to implement machine learning for video processing systems with a deep learning technique, especially with large datasets. As a result, the following tools are used to implement the suggested system:

a- Hardware

- Central Processing Unit (CPU): Intel(R) Core (TM) i7-10750H CPU @ 2.60GHz 2.59 GHz
- RAM: 16GB.
- Graphics Processing Unit (GPU): NVIDIA GetForce GTX 1660 TI 6GB.
- Hard Disk: 512 GB

b- Operating System

- Windows10, 64 bit

c- Programing Language

- Python 3.8, Jupyter, Colab site

4.3 Description of Dataset

The dissertation employs the datasets of Barack Obama stated in Section (3.2.1) to train and test the proposed system. The dataset of Barack Obama contains 83.14155 percent of the real videos and 16.85845 percent of the fake videos, as shown in Figure (4.1). The number of frames containing features is 14426, extracted from the datasets in the proposed system.

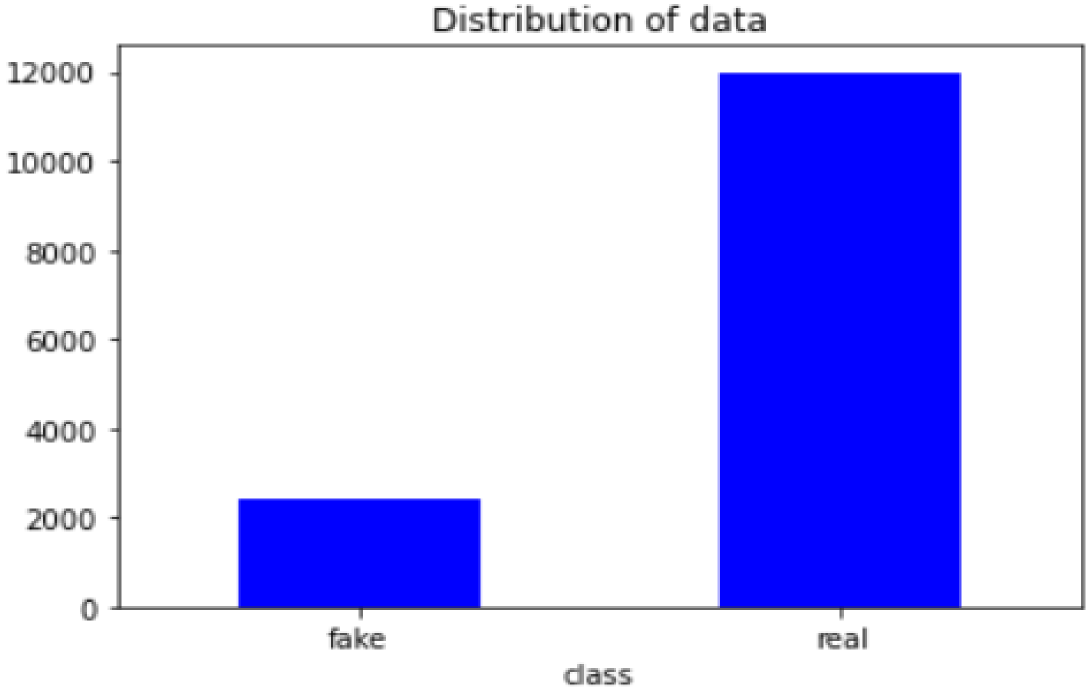


Figure (4.1): Distribution Dataset of Barack Obama

4.4 Results of Data Preprocessing

This stage is used for facial feature detection by resizing the frame to specific dimensions. When a dataset is entered into the method for the purpose of detecting features, many featureless frames (NaN) appear, as shown in Table (4.1). Frames without features are ignored. The main reason that these frames lack features is because of face occlusion. Sometimes, a small part of the face is visible, which is insufficient to calculate the action unit. As a result, some frames do not offer features, and the method ignores these frames.

Table (4.1): Some Featureless Frames from the Facial Action Unit and Pose

Id	AU01	AU02	AU04	AU06	AU07	AU10	AU12	AU14	AU15	AU17	AU23	AU24	Pitch	Roll	Yaw
1	0.104413	0.863236	0.037431	0.752315	0.721346	0.867614	0.998236	0.087883	0.184475	0.012468	0.015287	0.000628	-3.2784	-1.25691	30.30952
2	0.52162	0.727262	0.071662	0.711115	0.894822	0.062468	0.721545	0.728457	0.088015	0.303867	0.223858	0.00235	-0.05287	0.886075	30.57558
3	0.576822	0.321575	0.282381	0.590929	0.909182	0.131006	0.79189	0.722937	0.106982	0.187561	0.201432	0.001818	-22.7895	-17.0108	-4.83785
4	0.001943	0.490486	0.080512	0.992295	0.981327	0.136936	0.996286	0.249272	0.946056	0.002284	0.007708	1.57E-06	-29.2674	-14.0472	-7.54547
5	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
6	0.484652	0.103367	0.066312	0.393107	0.884752	0.085608	0.6859	0.532664	0.044943	0.043799	0.062336	0.000323	-16.2818	-8.30921	29.90874
7	0.15834	0.143277	0.086123	0.515931	0.531933	0.242743	0.911292	0.91766	0.06511	0.040331	0.006226	0.000126	-14.9357	-8.71978	25.37192
8	0.061438	0.178105	0.754572	0.868149	0.836029	0.144549	0.327301	0.738883	0.066634	0.154384	0.043275	0.01746	-4.06792	6.629462	40.29875
9	0.163512	0.411286	0.090386	0.679316	0.925412	0.011047	0.753102	0.592268	0.022852	0.011633	0.026958	0.000181	-15.9996	-8.94552	25.84165
10	0.316478	0.313731	0.029795	0.940084	0.929211	0.229055	0.928581	0.627105	0.031871	0.048937	0.012903	9.36E-05	-0.99441	-7.68112	-13.1361

4.5 Results of Feature Extraction

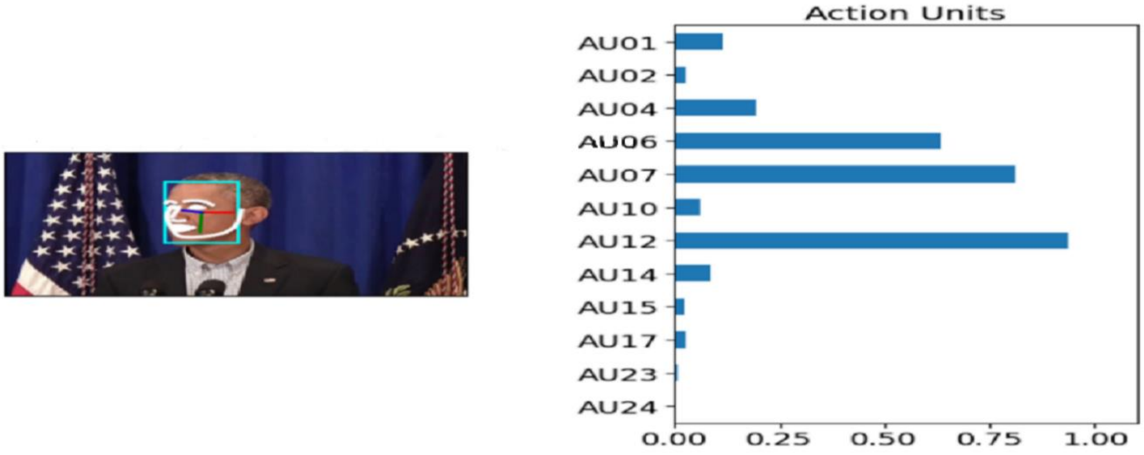
Facial features (action unit and face pose) are extracted for each frame, where the method reads one frame out of every twenty-four frames. As it extracts Fifteen features from each frame, it divides them into twelve action unit features and three face pose features. The network that extracts the action unit is JAA-Net, and the network that extracts the face pose is Hybrid Face Pose.

4.5.1 JAA-Net Model

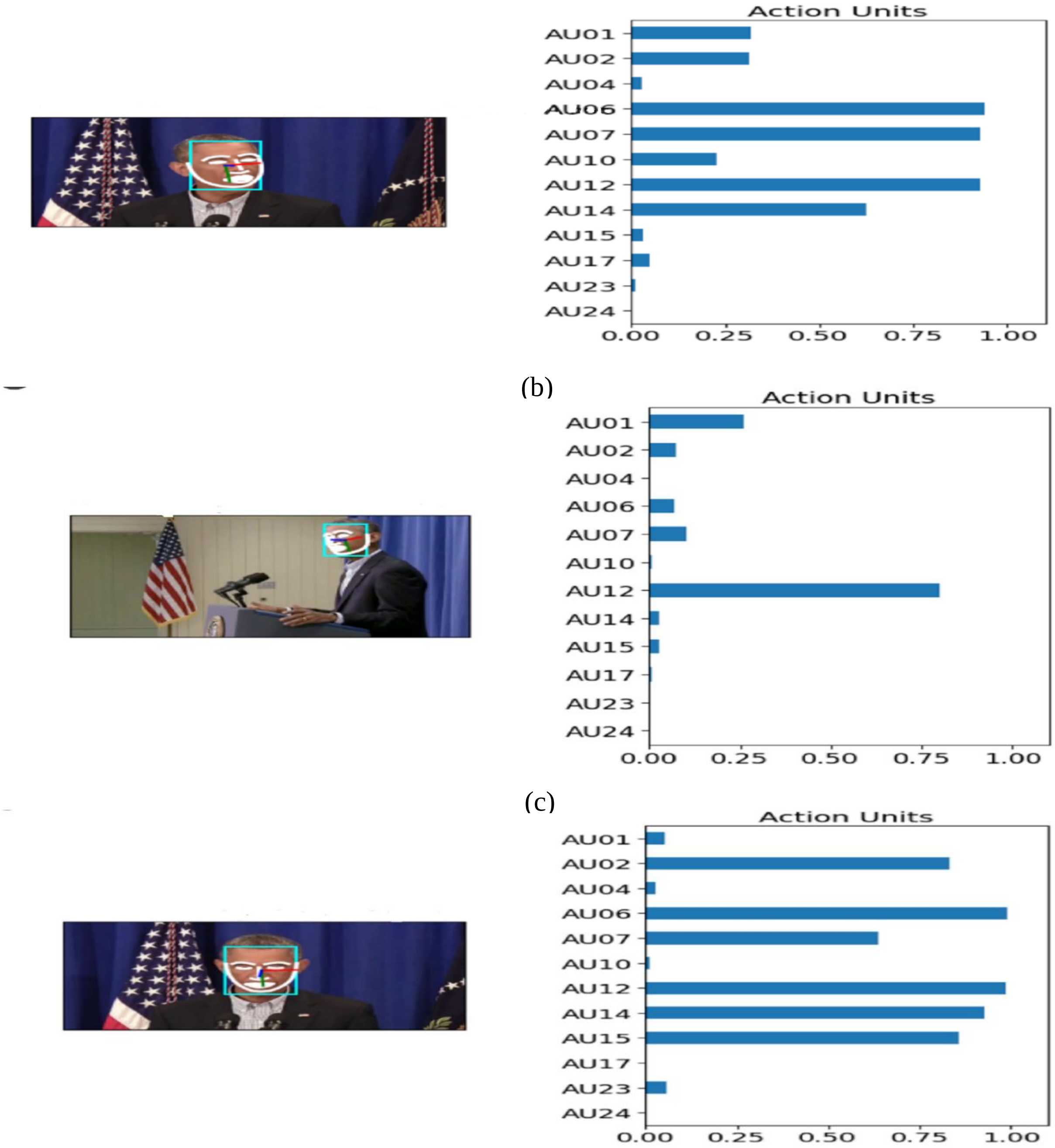
JAA-Net is used to extract the action unit, which is the basic component of facial expressions in emotional states. In typical situations, some action units are proportionally shared among several individuals. The proposed system is based on twelve action units. These are the most distinctive action units for facial expressions, such as sadness, anger, astonishment, and laughter, among others. The size of the dataset for Barack Obama is 227.97 minutes, so the number of frames is very large. One frame out of every twenty-four frames is read, and the reason behind choosing one frame out of every twenty-four is that these frames are close in action unit values. The output of the JAA-Net network is twelve action units for each frame entered into the network. These features are stored in records, as shown in Table (4.2). In this Table, ten frames of real video and ten frames of fake video are taken, for example.

The number of frames in which the action unit is recorded is 14381 frames. Each frame represents a person's behavior at a specific emotional moment. Each action unit has a different value in one frame and is repeated in subsequent frames according to the facial expression, as shown in Figure (4.2). Action unit values may be large in some frames and small in other frames, and this depends on the expressions and emotions represented by the person's behavior. There are some action units that are very repetitive in most facial expressions, and some of them occur rarely in some frames. The statistical distribution of the histogram and the incidence rates of each action unit can be seen as shown in Figure (4.3) It is observed in the graphs that action units 7 and 12 have a great value, and this indicates that these actions are repeated in many frames. The high frequency with which it occurs indicates that this action is involved in many emotional states. It is also noted in the action units 17, 23, and 24 that their value is low in many frames. The small value of these actions is due to their low occurrence in emotional states. It is an important feature in analyzing a person's behavior, and all of these features are important in behavior analysis.

A comparison of the types of action units using JAA-Net, logistic modeling, and SVM modeling is shown in Table (4.3). In this comparison, it is noted that each model depends on a specific set of action units, which represent a person's behavior in emotional states. JAA-Net is based on twelve types of action units, which represent the most distinctive behavior of people compared to models (logistics modeling and SVM modeling). Logistics modeling and SVM modeling contain action units that are common to most people's behavior. This action unit does not add information in order to predict real or fake videos.



(a)



(b)

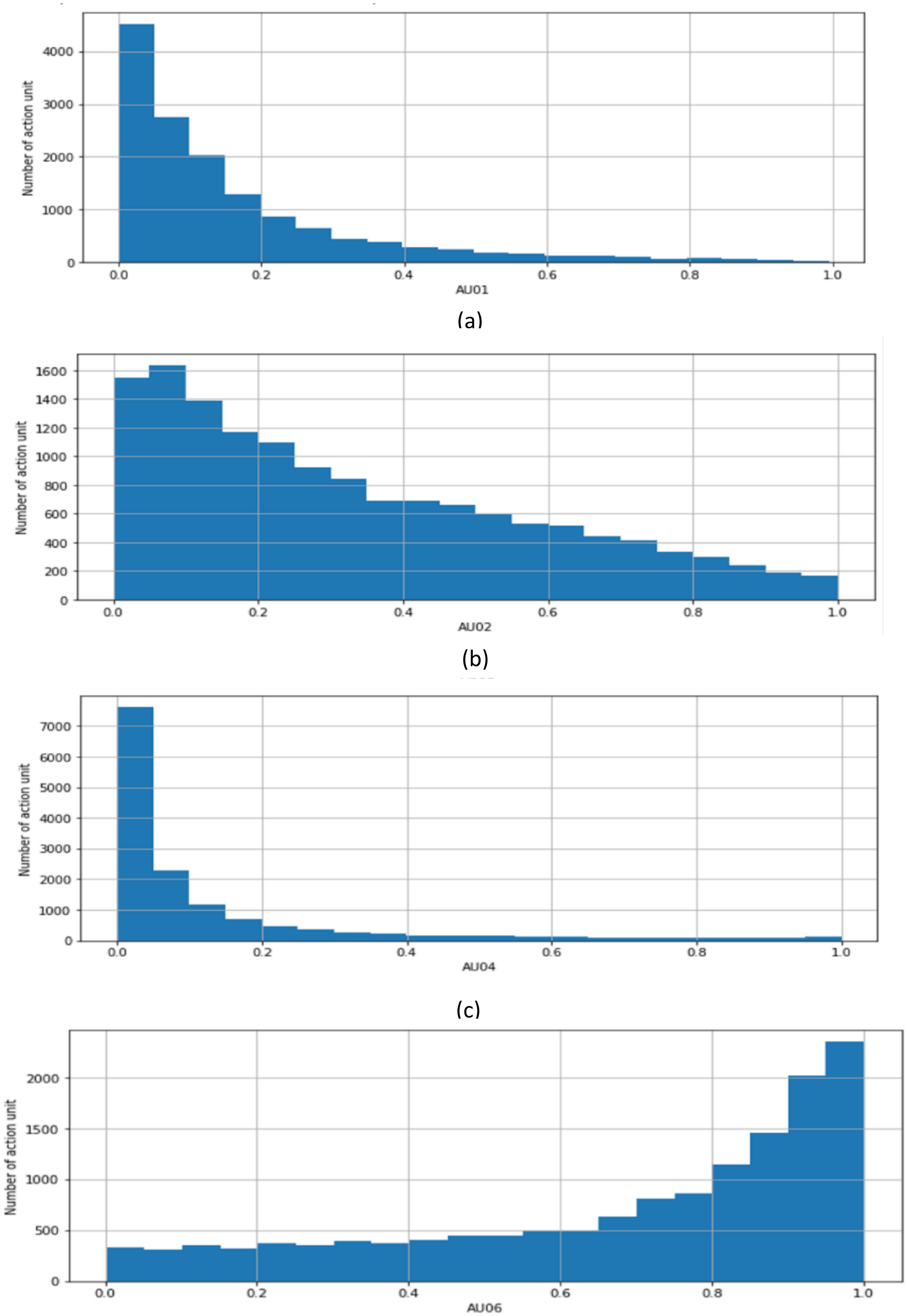
(c)

(d)

Figure (4.2): The Proportions of The Action Unit That Appear in The Face, In Addition to The Facial Expressions That Depend on The Movement of The Muscles, (a) Movement of the Face to The Left, (b) Movement of The Face to The Right, (c) The Lateral Aspect of The Face, (d) A Face-Down Movement

Table (4.2): The Action Unit Values that are Extracted from each Frame

AU01	AU02	AU04	AU06	AU07	AU10	AU12	AU14	AU15	AU17	AU23	AU24	class
0.063951	0.077517	0.128747	0.830497	0.875836	0.085770	0.985917	0.749140	0.436959	0.020913	0.018673	0.000079	Fake
0.063964	0.080328	0.064397	0.837139	0.881700	0.107231	0.990099	0.658891	0.352172	0.017647	0.015836	0.000075	Fake
0.021470	0.096209	0.027898	0.834912	0.922679	0.044449	0.996293	0.333420	0.098804	0.003405	0.026407	0.000037	Fake
0.040669	0.124593	0.047505	0.758507	0.867307	0.123618	0.993312	0.356017	0.124084	0.007178	0.007754	0.000030	Fake
0.013429	0.055261	0.121025	0.810898	0.889763	0.149019	0.995715	0.372219	0.172655	0.014591	0.009259	0.000051	Fake
0.204287	0.215976	0.052611	0.434404	0.848608	0.712056	0.994070	0.400216	0.065713	0.010062	0.039288	0.000051	Fake
0.198854	0.235322	0.060607	0.435450	0.855232	0.721436	0.993657	0.442716	0.066947	0.011054	0.035613	0.000050	Fake
0.177220	0.183314	0.055944	0.490405	0.858517	0.759331	0.994709	0.375909	0.096170	0.015805	0.033958	0.000106	Fake
0.158808	0.198933	0.049850	0.607992	0.743638	0.775094	0.996779	0.372415	0.047472	0.008768	0.045332	0.000098	Fake
0.171923	0.209367	0.149400	0.495804	0.806495	0.807258	0.996716	0.400435	0.064953	0.006442	0.018380	0.000069	Fake
0.049629	0.135138	0.008634	0.977036	0.989510	0.016730	0.977978	0.298869	0.115942	0.001579	0.001201	0.000002	Real
0.087020	0.266083	0.578985	0.856962	0.917598	0.007261	0.940133	0.713928	0.557505	0.005794	0.006416	0.000018	Real
0.027922	0.324587	0.149469	0.918475	0.951530	0.007347	0.928605	0.447587	0.455532	0.008542	0.014702	0.000005	Real
0.135980	0.192408	0.017735	0.843878	0.952870	0.004557	0.957094	0.365944	0.208264	0.002220	0.001772	0.000001	Real
0.172926	0.531866	0.017300	0.894080	0.941933	0.006579	0.971476	0.365242	0.450680	0.006607	0.005110	0.000005	Real
0.131580	0.494248	0.009843	0.936237	0.953196	0.009214	0.969020	0.343460	0.342838	0.006109	0.003635	0.000008	Real
0.036750	0.342757	0.219228	0.967212	0.955728	0.015860	0.972348	0.759643	0.729749	0.000783	0.003350	0.000001	Real
0.051639	0.219168	0.005860	0.939187	0.974634	0.018051	0.974668	0.415762	0.381708	0.004212	0.001406	0.000011	Real
0.082328	0.261664	0.164324	0.955870	0.973119	0.010861	0.961041	0.608194	0.381309	0.001395	0.002596	0.000001	Real
0.146618	0.530872	0.011614	0.897340	0.879659	0.010448	0.957534	0.602650	0.613416	0.003901	0.006918	0.000002	Real
0.049629	0.135138	0.008634	0.977036	0.989510	0.016730	0.977978	0.298869	0.115942	0.001579	0.001201	0.000002	Real

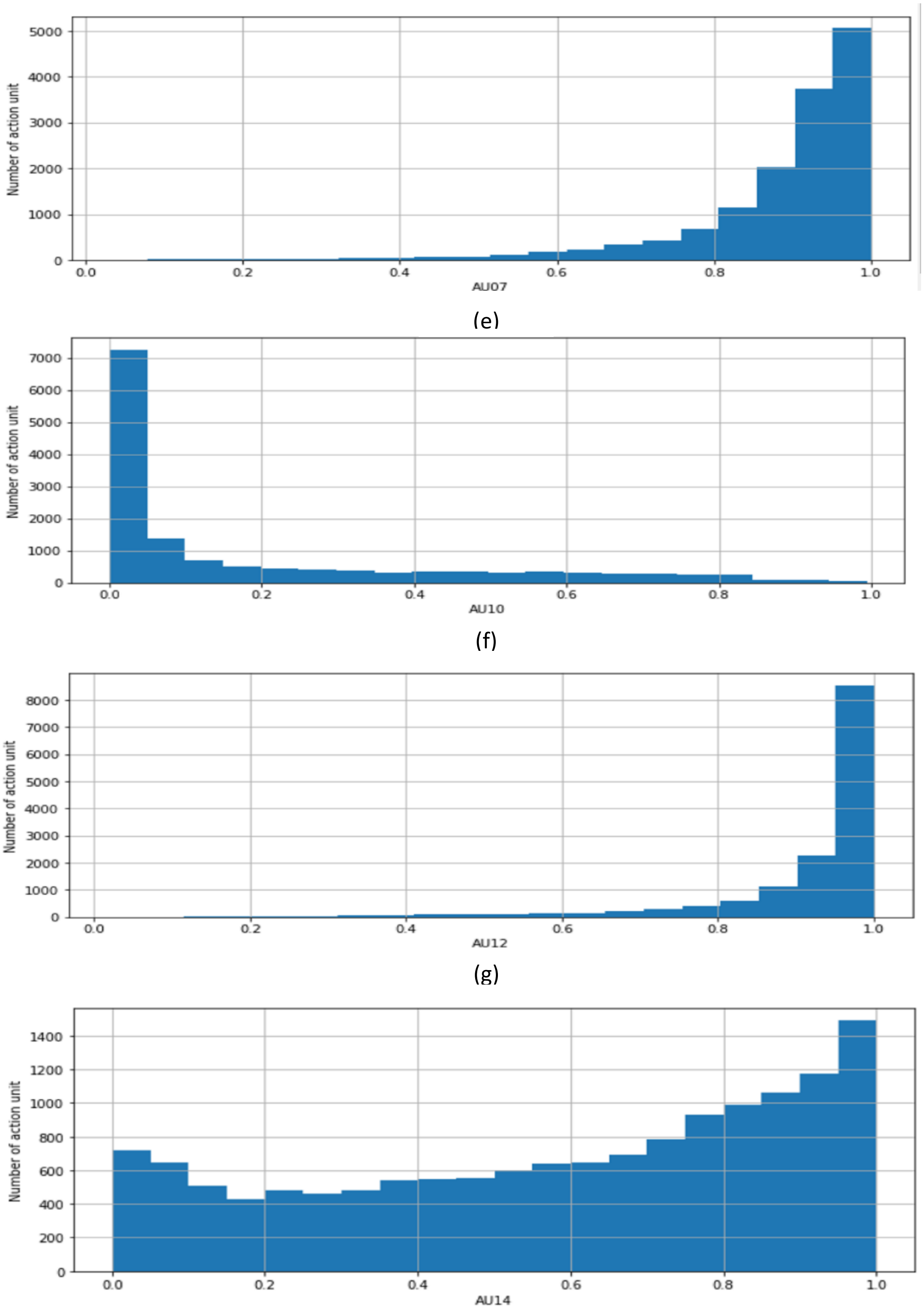

Number of action unit
4000
3000
2000
1000
0
0.0
0.2
0.4
0.6
0.8
1.0
AU01
Number of action unit
1600
1400
1200
1000
800
600
400
200
0
0.0
0.2
0.4
0.6
0.8
1.0
AU02
Number of action unit
7000
6000
5000
4000
3000
2000
1000
0
0.0
0.2
0.4
0.6
0.8
1.0
AU04
Number of action unit
2000
1500
1000
500
0
0.0
0.2
0.4
0.6
0.8
1.0
AU06

(a)

(b)

(c)

(d)

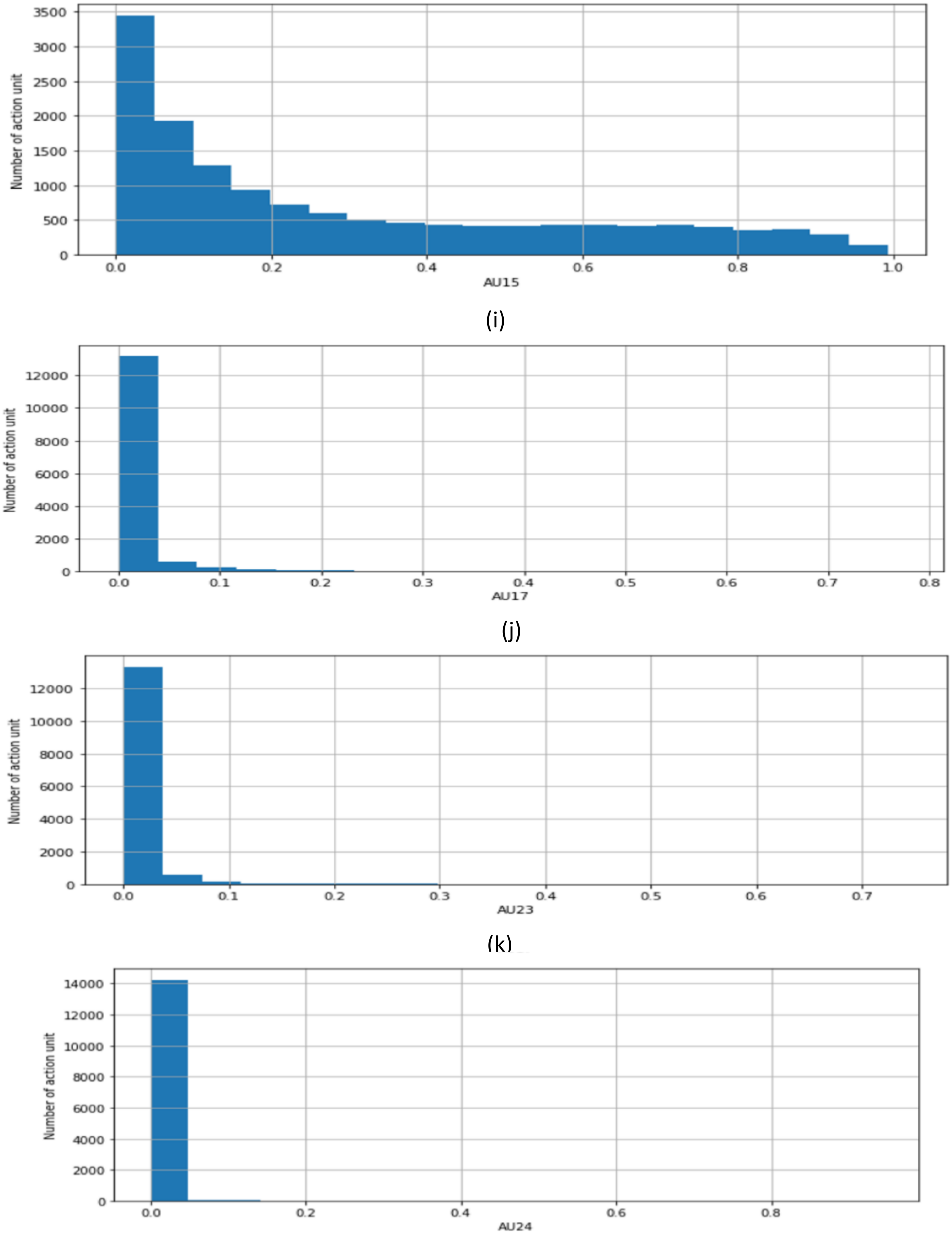

Number of action unit
5000
4000
3000
2000
1000
0
0.0
0.2
0.4
0.6
0.8
1.0
AU07
Number of action unit
7000
6000
5000
4000
3000
2000
1000
0
0.0
0.2
0.4
0.6
0.8
1.0
AU10
Number of action unit
8000
7000
6000
5000
4000
3000
2000
1000
0
0.0
0.2
0.4
0.6
0.8
1.0
AU12
Number of action unit
1400
1200
1000
800
600
400
200
0
0.0
0.2
0.4
0.6
0.8
1.0
AU14

(e)

(f)

(g)

(h)



(i)

(j)

(k)

(l)

Figure (4.3): Histogram Shows The Distribution and Repetition of The Action Unit in All Frames, (a) AU01 is Inner Brow Raiser,(b) AU02 is Outer Brow Raiser,(c) AU04 is Brow

Lowered,(d) AU06 is Cheek Raiser,(e) AU07 is Lid Tightener,(f) AU10 is Upper Lip Raiser,(g) AU12 is Lip Corner Puller,(h) AU14 is Dimpler,(i) AU15 is Lip Corner Depressor,(j) AU17 is Chin Raiser,(k) AU23 is Lip Tightener, and(l) AU24 is Lip Pressor

Table (4.3): The Types of Action Units (JAA-Net, Logistic, and SVM)

No.	JAA-Net	logistic	SVM	Facial Action Coding System
1.	AU01	AU01	AU01	Inner Brow Raiser
2.	AU02	AU02	AU02	Outer Brow Raiser
3.	AU04	AU04	AU04	Brow Lowered
4.		AU05	AU05	Upper Lid Raiser(surprise)
5.	AU06	AU06	AU06	Cheek Raiser
6.	AU07	AU07	AU07	Lid Tightener
7.		AU09	AU09	Nose Wrinkler
8.	AU10	AU10	AU10	Upper Lip Raiser
9.		AU11	AU11	Nasolabial Fold Deepener
10.	AU12	AU12	AU12	Lip Corner Puller
11.	AU14	AU14	AU14	Dimpler
12.	AU15	AU15	AU15	Lip Corner Depressor
13.	AU17	AU17	AU17	Chin Raiser
14.		AU20	AU20	Lip Stretcher
15.	AU23	AU23	AU23	Lip Tightener
16.	AU24	AU24	AU24	Lip Pressor
17.		AU25	AU25	Lips Part
18.		AU26	AU26	Jaw Drop
19.		AU28	AU28	Lip Suck
20.		AU43	AU43	Eyes Closure

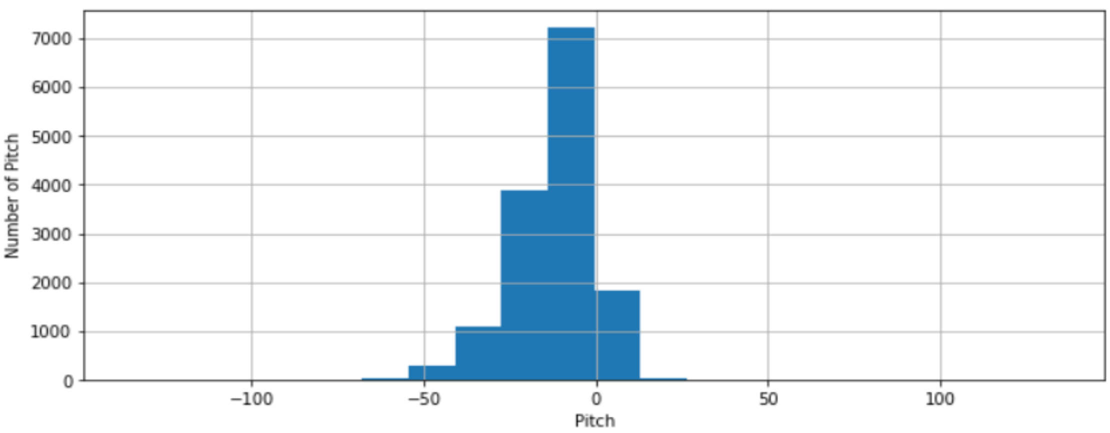
4.5.2 The Hybrid Face Pose

The same size as the dataset used in the previous step, the size of the dataset for Barack Obama is 227.97 minutes. The output of the network is three face pose features for each frame entered into the network. These features are stored in records, as shown in Table (4.4). In this Table, ten frames of real video and ten frames of fake video are taken, for example. The angle at which the head moves left and right is known as yaw (rotation around the Y-axis). The yaw angle feature is more prominent than the rest of the features, as shown in Figure (4.4). The reason is that Barack Obama, in his kinematic behavior, has a lot of facial poses left and right.

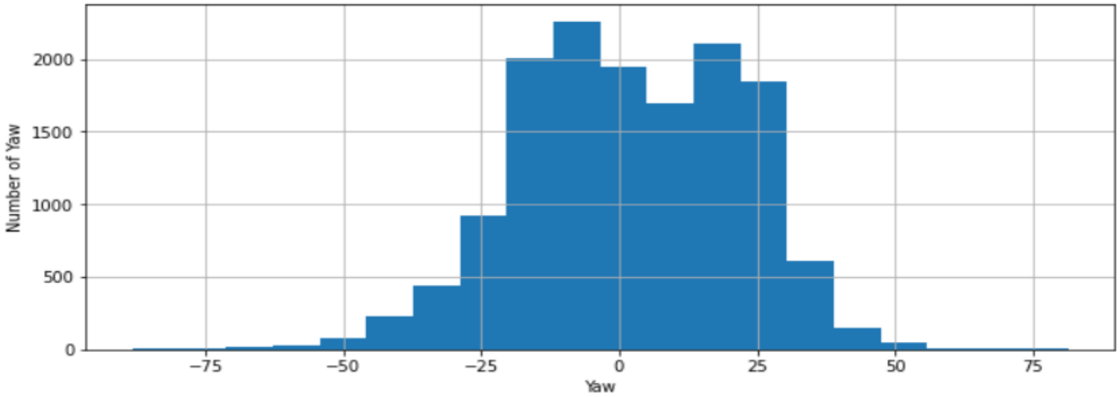
Table (4.4): The Face Pose Values that are Extracted from each Frame

Pitch	Roll	Yaw	class
-3.247937566	-4.411995925	-17.0691136	Fake
-13.57329167	-1.106462467	-15.30112384	Fake
-4.937920379	-1.723398503	-16.23660041	Fake
-5.233145872	-2.002225594	-14.29503054	Fake
-5.64409766	-5.145593958	-19.58819927	Fake
-9.137226253	-0.553054397	-14.18559441	Fake
-5.813105338	-0.363239955	-14.27583297	Fake
-2.753507042	-3.706595138	-11.25936282	Fake
-1.361913952	-3.24716221	-7.439307537	Fake
-2.596701644	-0.425416226	-4.51408786	Fake
-35.43127721	-8.432055768	-23.40144442	Real
-10.95924001	-4.714812918	22.19759342	Real
-41.6718828	-13.17878492	-14.32016993	Real

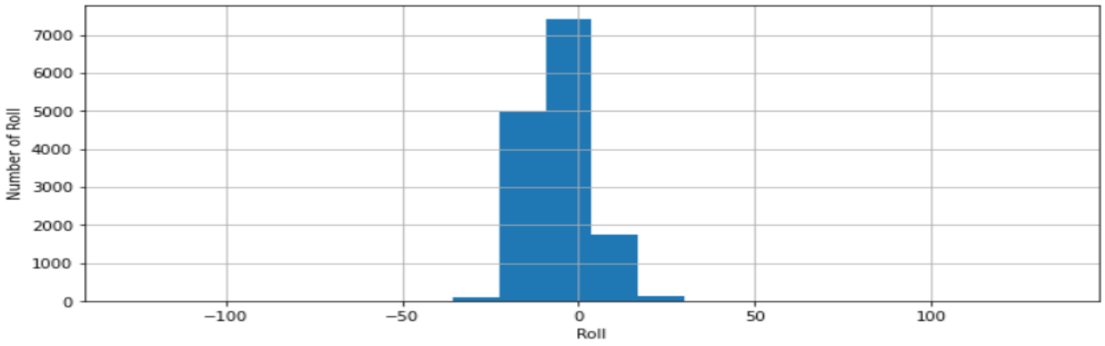
-39.38036341	-11.23380072	-13.271737	Real
-2.661374508	-4.816435319	21.07569801	Real
-31.27399862	-13.49656601	-9.080678121	Real
-41.69868865	-9.840846534	-17.70546188	Real
-9.424504586	-8.346835714	-11.40830333	Real
-47.12117788	-8.895422898	-16.64500724	Real
-47.05294375	-9.129821511	-18.03101862	Real
0.049629	0.135138	0.008634	Real



a



b



c

Figure (4.4): Histogram Shows the Distribution and Repetition of the Face Pose in all Frames, (a) the Distribution of Pitch Feature, (b) the Distribution of Yaw Feature, and(b) the Distribution of Roll Feature

4.6 BiLSTM Prediction

After extracting the features from each frame of the Barack Obama dataset videos, the features are stored in a profile. This profile contains all the extracted features (action unit and face pose). The number of entries is 14426, and the number of features is Fifteen, in addition to the label as shown in Table (4.5). In this Table, ten frames of real video and ten frames of fake video are taken, for example. The process of labeling the constraints involves extracting the features from the videos. When features are extracted from real videos, the label will be (1), and for fake videos, the label will be (0). The profile file that is extracted from Barack Obama's dataset, which contains real and fake data, is entered into the BiLSTM network for training. The number of epochs used in the prediction process is 100, and this helped in giving good results.

Table (4.5): Barack Obama's Profile was Extracted from each Frame

AU01	AU02	AU04	AU06	AU07	AU10	AU12	AU14	AU15	AU17	AU23	AU24	Pitch	Roll	Yaw	class
0.104413	0.863236	0.037431	0.752315	0.721346	0.867614	0.998236	0.087883	0.184475	0.012468	0.015287	0.000628	-3.2784	-1.25691	30.30952	**1**
0.52162	0.727262	0.071662	0.711115	0.894822	0.062468	0.721545	0.728457	0.088015	0.303867	0.223858	0.00235	-0.05287	0.886075	30.57558	**1**
0.576822	0.321575	0.282381	0.590929	0.909182	0.131006	0.79189	0.722937	0.106982	0.187561	0.201432	0.001818	-22.7895	-17.0108	-4.83785	**1**
0.001943	0.490486	0.080512	0.992295	0.981327	0.136936	0.996286	0.249272	0.946056	0.002284	0.007708	1.57E-06	-29.2674	-14.0472	-7.54547	**1**
0.11016	0.059487	0.065054	0.603617	0.838806	0.110434	0.88352	0.291323	0.014987	0.042948	0.015676	0.000124	1.413898	4.303543	26.02552	**1**
0.484652	0.103367	0.066312	0.393107	0.884752	0.085608	0.6859	0.532664	0.044943	0.043799	0.062336	0.000323	-16.2818	-8.30921	29.90874	**1**
0.15834	0.143277	0.086123	0.515931	0.531933	0.242743	0.911292	0.91766	0.06511	0.040331	0.006226	0.000126	-14.9357	-8.71978	25.37192	**1**
0.061438	0.178105	0.754572	0.868149	0.836029	0.144549	0.327301	0.738883	0.066634	0.154384	0.043275	0.01746	-4.06792	6.629462	40.29875	**1**
0.163512	0.411286	0.090386	0.679316	0.925412	0.011047	0.753102	0.592268	0.022852	0.011633	0.026958	0.000181	-15.9996	-8.94552	25.84165	**1**
0.316478	0.313731	0.029795	0.940084	0.929211	0.229055	0.928581	0.627105	0.031871	0.048937	0.012903	9.36E-05	-0.99441	-7.68112	-13.1361	**1**
0.06395111	0.07751657	0.12874737	0.83049726	0.87583625	0.08576966	0.98591727	0.74914	0.43695882	0.020912504	0.018673405	7.89E-05	-3.247937566	-4.411995925	-17.0691136	**0**
0.06396372	0.08032804	0.064396694	0.83713853	0.8816998	0.10723137	0.99009925	0.65889144	0.35217178	0.017647369	0.015836306	7.53E-05	-13.57329167	-1.106462467	-15.30112384	**0**
0.021470033	0.09620921	0.02789804	0.8349115	0.92267853	0.044448968	0.9962925	0.33341965	0.098804094	0.003404864	0.02640741	3.71E-05	-4.937920379	-1.723398503	-16.23660041	**0**
0.04066936	0.12459276	0.047504716	0.7585072	0.8673068	0.1236176	0.99331164	0.3560169	0.12408415	0.007178119	0.007753876	2.99E-05	-5.233145872	-2.002225594	-14.29503054	**0**
0.013429074	0.05526149	0.12102456	0.81089807	0.8897632	0.14901859	0.99571496	0.37221894	0.17265514	0.014590863	0.009259183	5.09E-05	-5.64409766	-5.145593958	-19.58819927	**0**
0.2042872	0.21597561	0.05261073	0.43440443	0.8486079	0.7120564	0.99406976	0.40021616	0.06571277	0.010062032	0.039288364	5.08E-05	-9.137226253	-0.553054397	-14.18559441	**0**
0.19885425	0.23532207	0.060607053	0.4354502	0.855232	0.7214363	0.9936574	0.44271633	0.066947214	0.0110538	0.03561274	5.01E-05	-5.813105338	-0.363239955	-14.27583297	**0**
0.17722037	0.18331413	0.05594395	0.490405	0.85851717	0.75933117	0.994709	0.37590888	0.09616956	0.015804794	0.03395775	0.000105808	-2.753507042	-3.706595138	-11.25936282	**0**
0.15880753	0.19893254	0.04985046	0.6079918	0.7436377	0.77509385	0.9967794	0.3724152	0.047472246	0.008767581	0.045331858	9.77E-05	-1.361913952	-3.24716221	-7.439307537	**0**
0.17192297	0.20936745	0.14940019	0.49580446	0.8064947	0.8072582	0.99671614	0.40043467	0.06495327	0.006441537	0.018379755	6.86E-05	-2.596701644	-0.425416226	-4.51408786	**0**

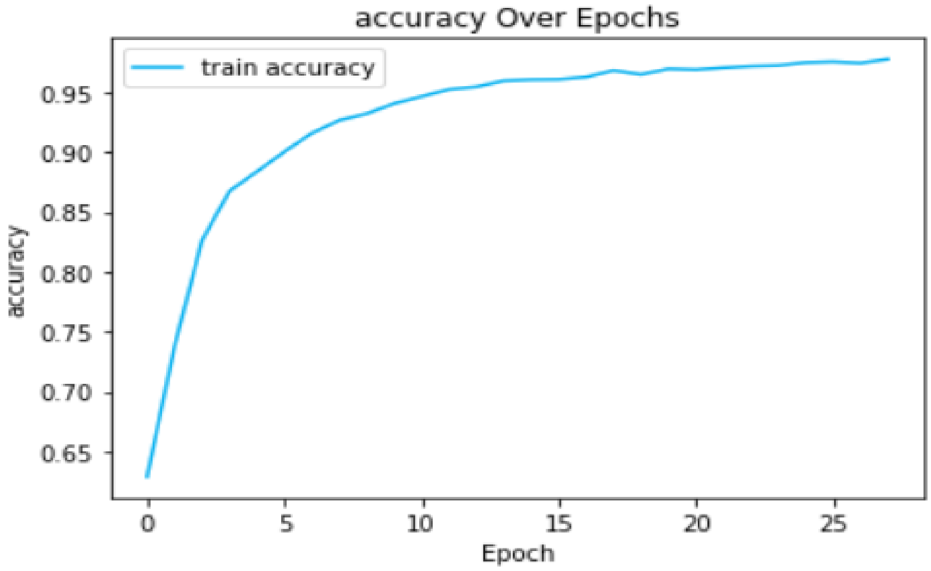
- **Training Mode:**

In the training phase, the dataset is divided into training data and test data based on k-fold values. A k-fold value is (5,6), which in turn divided the training and testing data. The k-fold value that is chosen is the best of K in terms of giving good results. Before the training process, the data in the imbalanced profile is processed using the SMOTE function. The accuracy and loss of the model after the training are shown in Table (4.6). The higher the k-fold values, the slighter improvement occurs in the evaluation measure in the method. The value of k-fold when it is equal to five or equal to six is the best value because it gives the best results in the model.

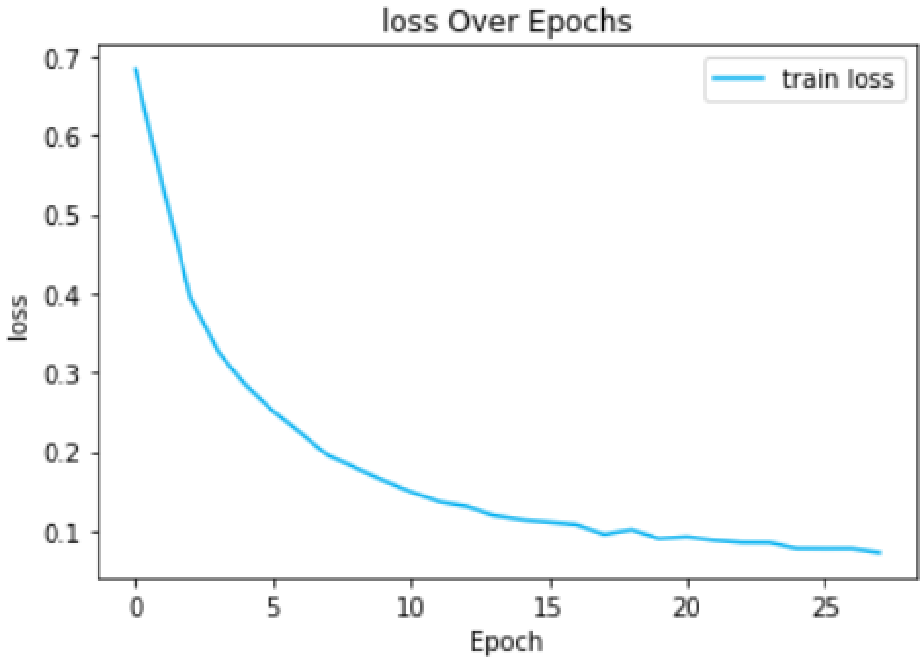
Table (4.6): Accuracy and Loss Metrics in the Training the Model

k-fold		Accuracy	loss
K=2	Maximum	96.531%	0.22745%
	Minimum	91.662%	0.10742%
	Overall	94.097%	0.16744%
K=5	Maximum	99.635%	0.03364%
	Minimum	99.171%	0.01881%
	Overall	99.521%	0.02196%
K=6	Maximum	99.169%	0.07534%
	Minimum	97.708%	0.03465%
	Overall	98.894%	0.04185%
K=7	Maximum	98.891%	0.09727%
	Minimum	96.755%	0.04379%
	Overall	98.555%	0.05205%

During the training process, the training data and test data are set based on the K-fold value. When the value of K-fold is equal to five and the epoch is 100, then the total process of the epoch is 100 × 5, equal to 500 epochs. Each fold is taken and trained for 100 epochs so that the training accuracy curve and loss curve reach a steady state, as shown in Figure (4.5). When comparing the accuracy and loss measures, there is a difference, and this depends on the epoch and K-fold values. The accuracy and loss curves are generally good in the training phase, as the accuracy values gradually increase and the loss values also decrease gradually. As for the validation set, they are the result of splits of the dataset depending on the K-fold values, as described previously. The curves of the training accuracy measure and the validation accuracy measure show that they are convergent, as shown in the Figure (4.6), Figure (4.7).



(a)



(b)

Figure (4.5): Training Accuracy and Loss when the K-Fold Size Equals 5, (a) the Accuracy Curve of Training, (b) the Loss Curve of Training

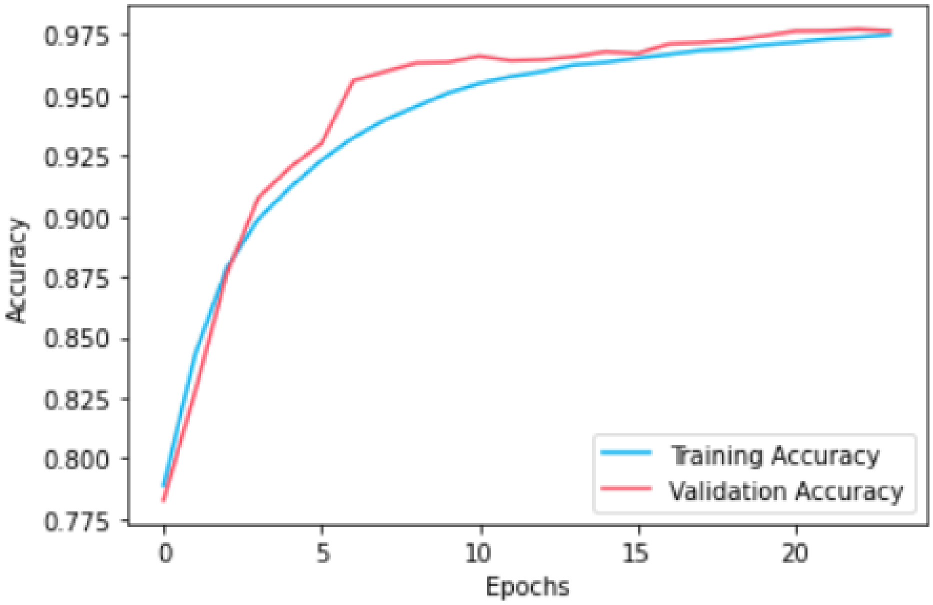
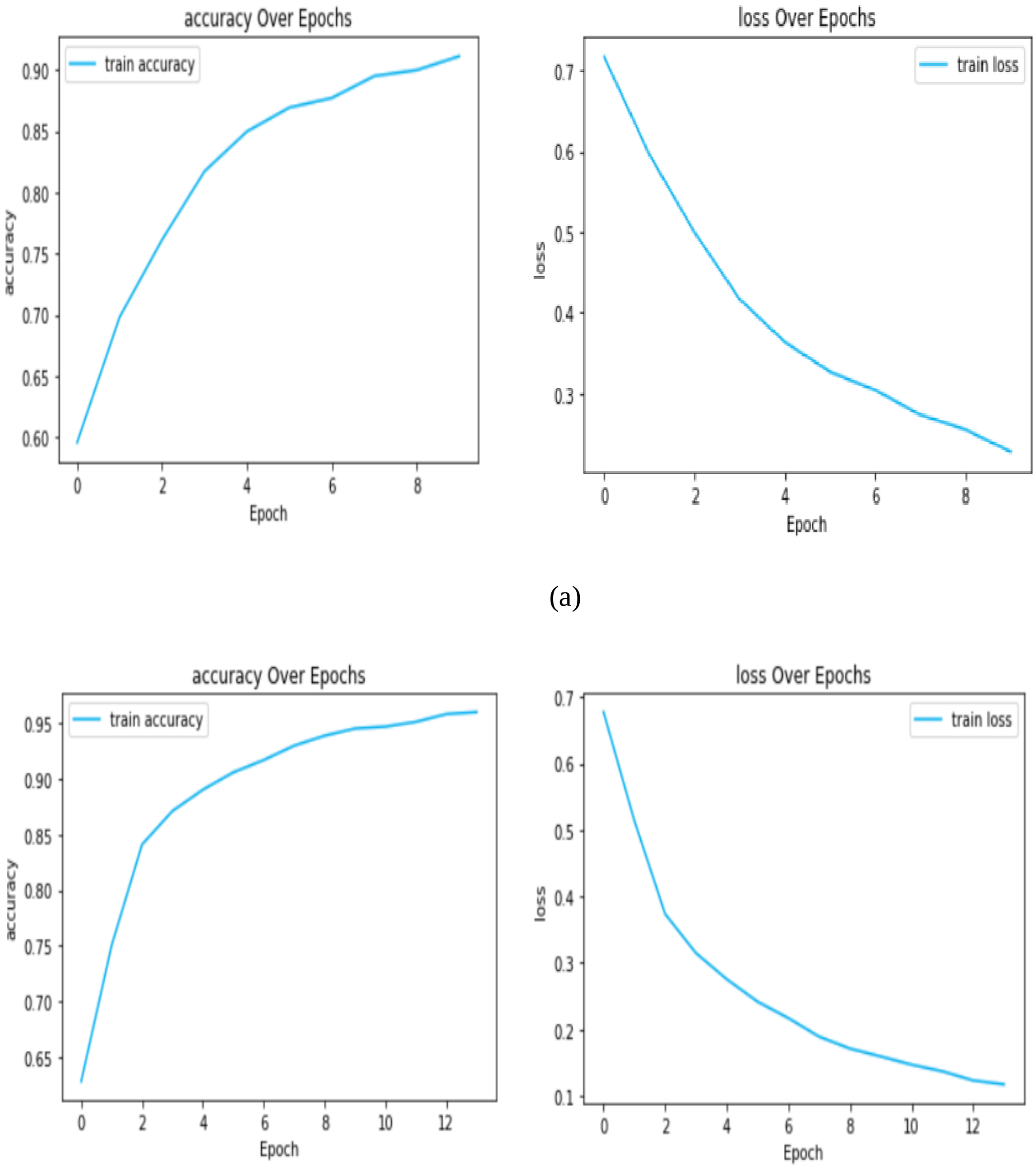
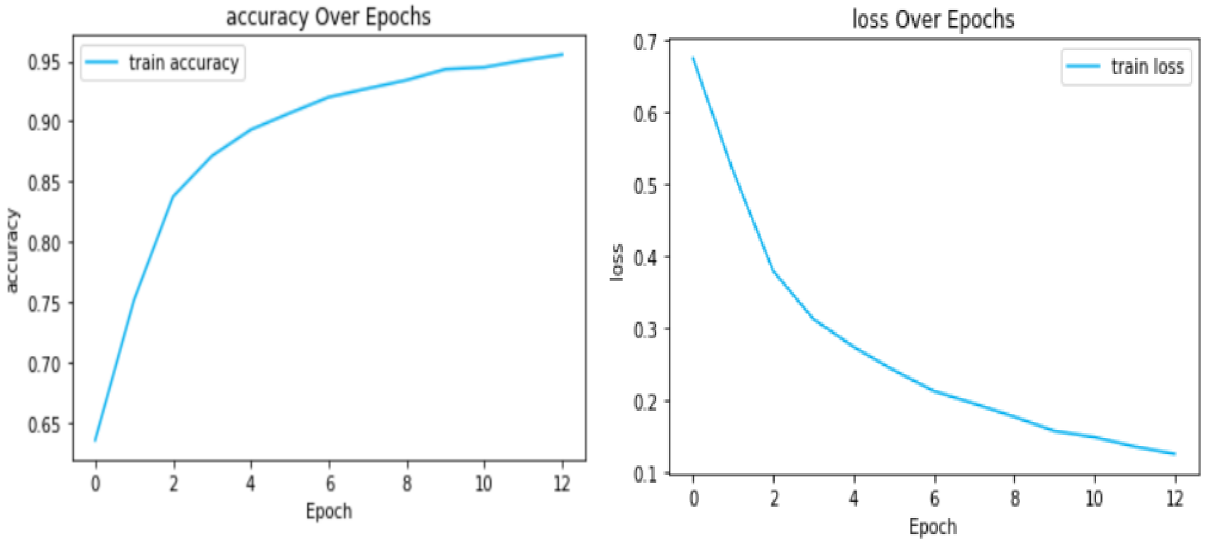


Figure (4.6): Convergence of Training Accuracy and Validation Accuracy Measures


accuracy Over Epochs
train accuracy
accuracy
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0
2
4
6
8
Epoch
loss Over Epochs
train loss
loss
0.7
0.6
0.5
0.4
0.3
0
2
4
6
8
Epoch
accuracy Over Epochs
train accuracy
accuracy
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0
2
4
6
8
10
12
Epoch
loss Over Epochs
train loss
loss
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
2
4
6
8
10
12
Epoch

(a)

(b)



(c)

Figure (4.7): The Training Accuracy Phase and Loss Using Different Values of K-Fold,(a) the Training Accuracy Phase and Loss Using The Epoch Size Equals 100 and the K-fold Size Equals 2 ,(b) the Training Accuracy Phase and Loss Using The Epoch Size Equals 100 and the K-fold size equals 6,(c) the Training Accuracy Phase and Loss Using The Epoch Size Equals 100 and the K-fold Size Equals 7

- **Testing Mode:**

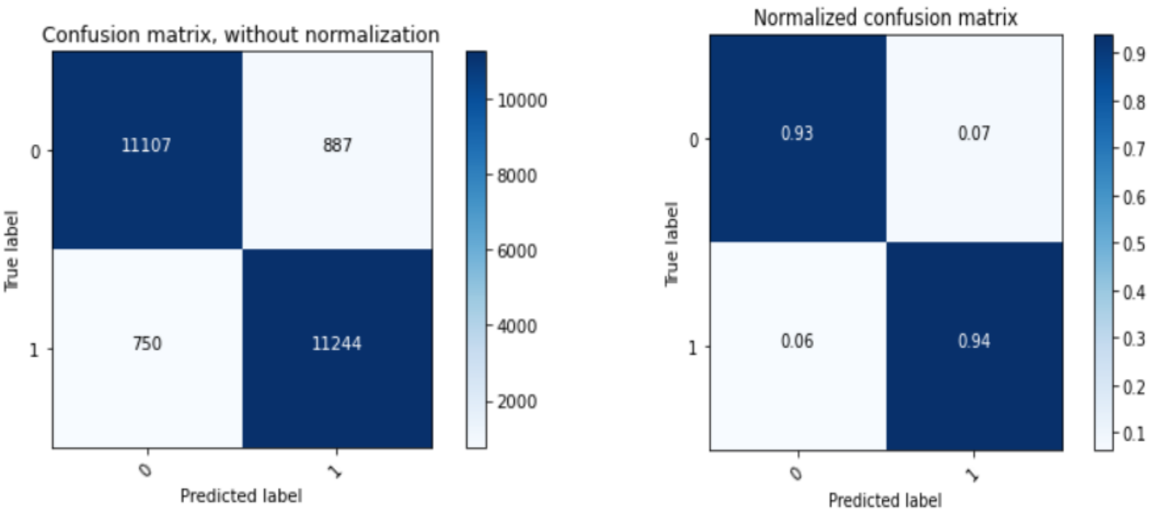
To evaluate the model, accuracy, recall, support, and F1 score are used, as shown in Table (4.7). Predictions using test data and different values of k-fold cross-validation are used. The experiments conducted on the method using different values of the k-fold cross-validation found that the best results can be obtained when the k-fold cross-validation values are equal to 5. The accuracy value and the standard deviation of the model's general accuracy value (average) using the test values are also high when the k-fold cross-validation values are equal to 5, as shown in Table (4.8). The confusion matrix shows the prediction rates, so the number of misclassifications is also very low when the k-fold cross-validation values are equal to 5, as shown in Figure (4.8).

Table (4.7): Performance Metrics for the Model

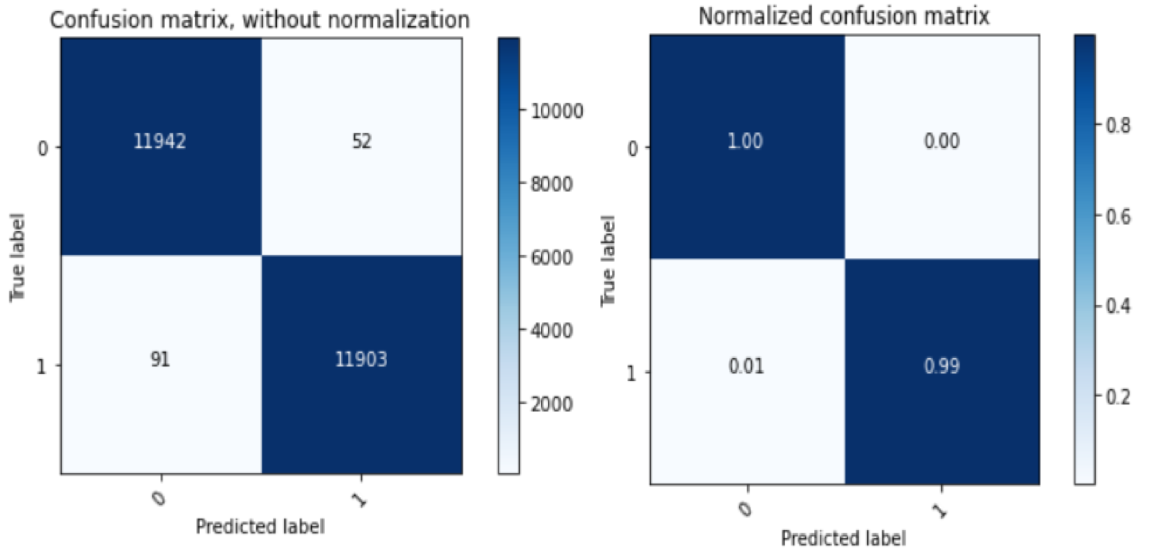
k-fold	Type class	precision	Recall	F1-score	support	accuracy
K=2	Fake	93%	94%	93%	11857	93%
	Real	94%	93%	93%	12131	93%
K=5	Fake	100%	99%	99%	12033	99%
	Real	99%	100%	99%	11955	99%
K=6	Fake	99%	98%	99%	12113	99%
	Real	98%	99%	99%	11875	99%
K=7	Fake	99%	98%	99%	12067	98%
	Real	98%	99%	98%	11921	98%

Table (4.8): The Accuracy and Standard Deviation of Accuracy Value

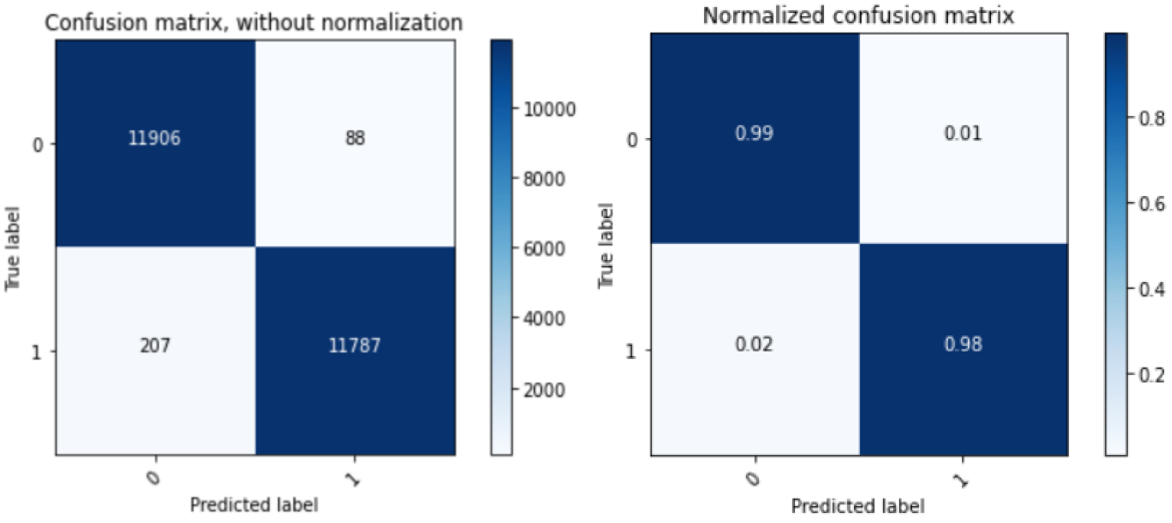
k-fold	value	
K=2	Mean accuracy	93.175%
	Standard deviation of accuracy	0.02030%
K=5	Mean accuracy	**99.403%**
	Standard deviation of accuracy	**0.00711%**
K=6	Mean accuracy	98.770%
	Standard deviation of accuracy	0.00914%
K=7	Mean accuracy	98.478%
	Standard deviation of accuracy	0.00895%



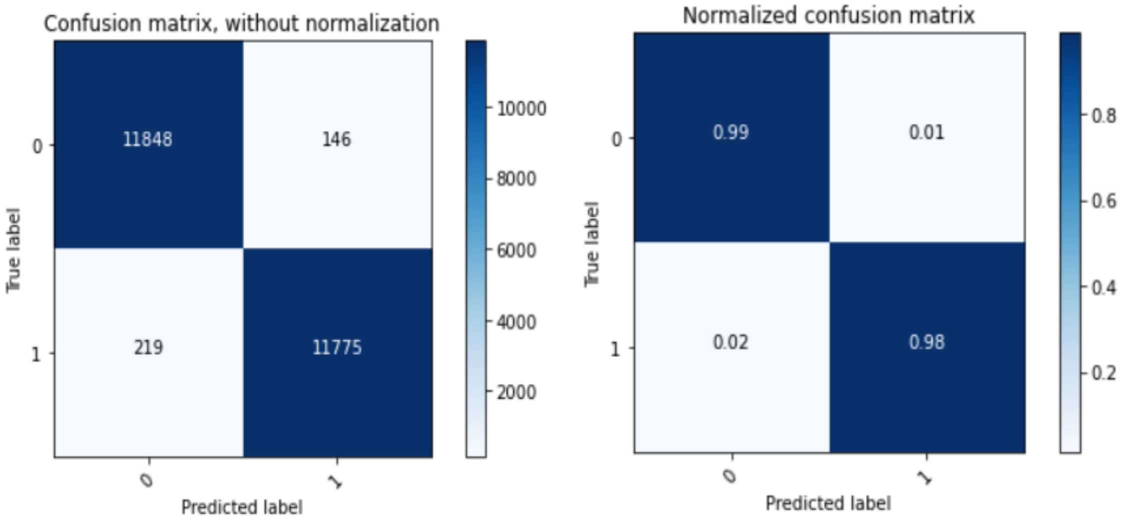
(a)


Confusion matrix, without normalization
11942
52
91
11903
True label
0
1
Predicted label
0
1
10000
8000
6000
4000
2000
Normalized confusion matrix
1.00
0.00
0.01
0.99
True label
0
1
Predicted label
0
1
0.8
0.6
0.4
0.2

(b)


Confusion matrix, without normalization
11906
88
207
11787
True label
0
1
Predicted label
0
1
10000
8000
6000
4000
2000
Normalized confusion matrix
0.99
0.01
0.02
0.98
True label
0
1
Predicted label
0
1
0.8
0.6
0.4
0.2

(c)



(d)

Figure (4.8): The Confusion Matrix with Normalized and Without Normalized Shows The Prediction Rates, (a) Confusion Matrix when k-Fold Cross-Validation is 2, (b) Confusion Matrix when k-Fold Cross-Validation is 5,(c) Confusion Matrix when k-Fold Cross-Validation is 6,(d) Confusion Matrix when k-Fold Cross-Validation is 7

4.7 Comparison of BiLSTM Results with Other Methods for Feature Extraction

Some methods have been used to extract features from the Barack Obama dataset. The proposed system utilized several important methods, including logistics, SVM, JAA-Net, and face pose. Each of the two methods is combined, and experiments prove that some gave good results and others gave low results. Therefore, the results of these methods are presented in the following sections:

4.7.1 Bil LSTM Prediction with the Profile Created Using Only JAA-Net (Without Face Pose)

Use a profile created using the JAA-Net model without the face pose features in the prediction model. JAA-Net model extracts twelve features (action units) as in

Section (4.5.1). The number of frames that contain action unit features is 14381. The results are lower than the features extracted by JAA-Net and face pose.

- **Training Mode:**

In the training phase, the dataset is divided into training and test data based on k-fold values, as in section (4.6). The accuracy and loss results of the model using only the action unit features are lower than those extracted by JAA-Net and face pose, as shown in Table (4.9).

Table (4.9): Accuracy and Loss Metrics in the Training the Model

k-fold=5	Accuracy	loss
Maximum	97.6364 %	0.12980 %
Minimum	95.317 %	0.07537 %
Overall	97.102 %	0.08717 %

- **Testing Mode:**

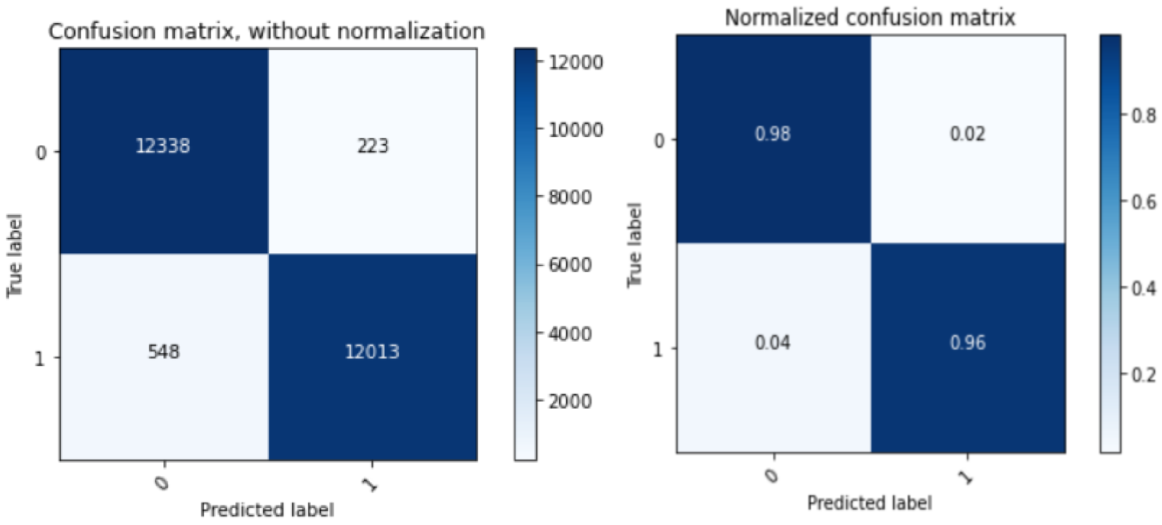
In the model testing phase, the measures accuracy, recall, support, and F1 score are less than the measures of the JAA-Net and face pose model together, as shown in Table (4.10). The model prediction is less compared to the profile that was extracted together by (JAA-Net and face pose). The accuracy measures and the standard deviation of the general accuracy value of the model are also less as shown in Table (4.11). The confusion matrix shows the prediction rates, and therefore the number of misclassification cases is greater compared to the JAA-Net and face pose, as shown in Figure (4.9).

Table (4.10): Performance Metrics for the Model

k-fold	Type class	precision	Recall	F1-score	support	accuracy
K=5	Fake	98%	96%	97%	12886	97%
	Real	96%	98%	97%	12236	97%
average		97%	97%	97%		97%

Table (4.11): The Accuracy and Standard Deviation of the Accuracy Value

k-fold=5	value
Mean accuracy	96.931%
Standard deviation of accuracy	0.01372%



(a) (b)

Figure (4.9): Confusion Matrix for the Profile Generated with JAA-Net, (a) The Confusion Matrix Without Normalized, (b) The Confusion Matrix with Normalized

4.7.2 BiLSTM Prediction with The Profile Created Using Only Face Pose

For the prediction model, use only the profiles created with face pose modeling. Face pose modeling is used to extract three features (face pose) as in section (4.5.2). The number of frames containing the features of the procedure face pose (14426). The results showed less compared to the features extracted together by JAA-Net and face pose.

- **Training Mode:**

The accuracy and loss results of the model using only the face pose features are lower than those extracted by JAA-Net and face pose, as shown in Table (4.12).

Table (4.12): Accuracy and Loss Metrics in the Training the Model

k-fold=5	Accuracy	loss
Maximum	82.136%	0.43061%
Minimum	79.072%	0.40123%
Overall	81.358%	0.40933%

- **Testing Mode:**

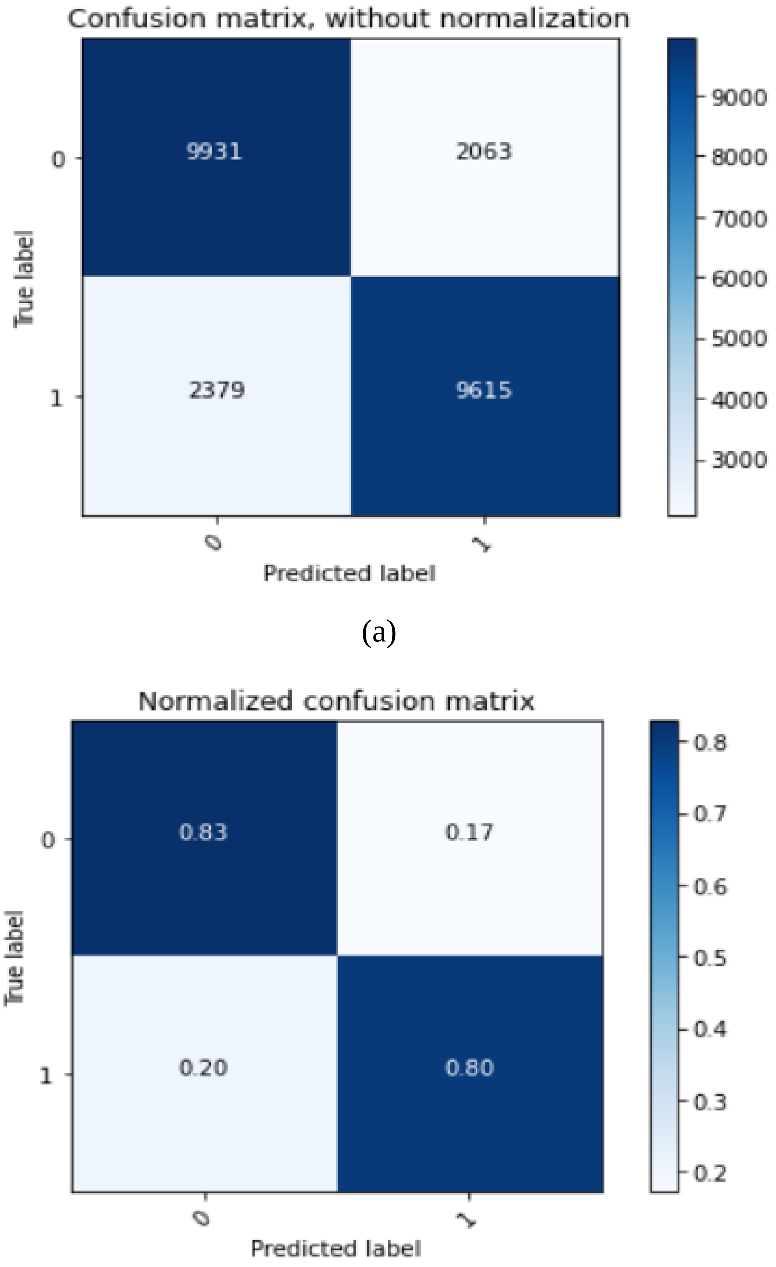
In the model testing phase, the measures are lower than those of JAA-Net and the face pose model together, as shown in table (4.13). The model prediction is less compared to the profile that was extracted together by (JAA-Net and face pose). The accuracy measures and the standard deviation of the general accuracy value of the method are also less as shown in Table (4.14). The confusion matrix shows the prediction rates, and therefore the number of misclassification cases is greater compared to the JAA-Net and face pose, as shown in Figure (4.10).

Table (4.13): Performance Metrics for the Model

k-fold	Type class	precision	Recall	F1-score	support	accuracy
K=5	Fake(0)	83%	81%	82%	12310	81%
	Real(1)	80%	82%	81%	11678	81%
average		81%	81%	81%		81%

Table (4.14): The Accuracy and Standard Deviation of the Accuracy Value

k-fold=5	value
Mean accuracy	81.482%
Standard deviation of accuracy	0.01153%



(a)

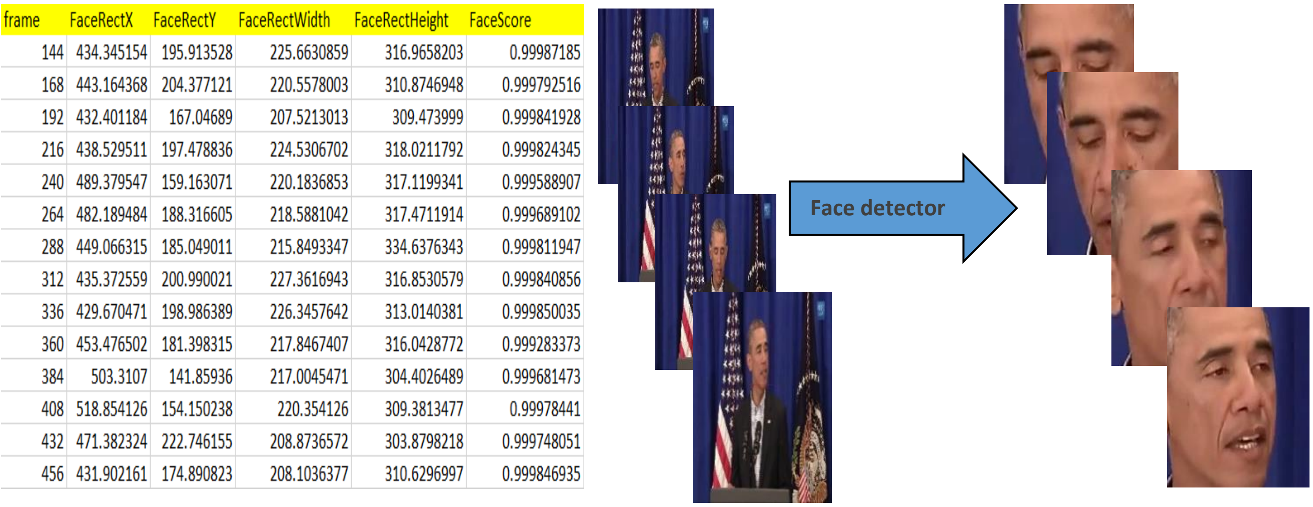
(b)

Figure (4.10): Confusion Matrix for the Profile Generated with Face Pose, (a) The Confusion Matrix Without Normalized, (b) The Confusion Matrix with Normalized

4.7.3 BiLSTM Prediction with the Profile Created Using Logistic

Creating a profile using logistics modeling, depends on a set of steps. The first step is face detection; frames of videos that do not contain a face are excluded.

In the face detection step, all videos (the dataset) are sent to the method, as shown in Figure (4.11).

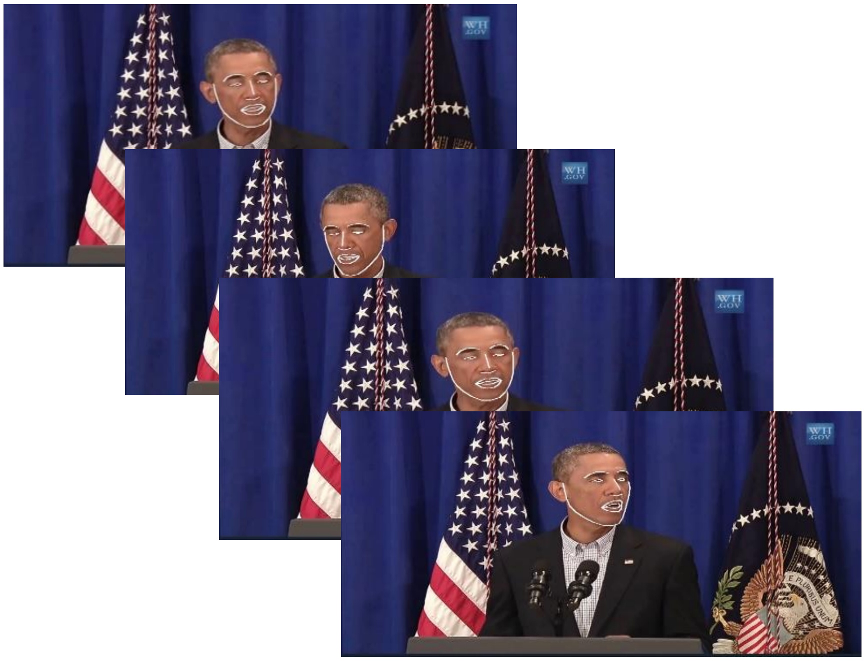


frame	FaceRectX	FaceRectY	FaceRectWidth	FaceRectHeight	FaceScore
144	434.345154	195.913528	225.6630859	316.9658203	0.99987185
168	443.164368	204.377121	220.5578003	310.8746948	0.999792516
192	432.401184	167.04689	207.5213013	309.473999	0.999841928
216	438.529511	197.478836	224.5306702	318.0211792	0.999824345
240	489.379547	159.163071	220.1836853	317.1199341	0.999588907
264	482.189484	188.316605	218.5881042	317.4711914	0.999689102
288	449.066315	185.049011	215.8493347	334.6376343	0.999811947
312	435.372559	200.990021	227.3616943	316.8530579	0.999840856
336	429.670471	198.986389	226.3457642	313.0140381	0.999850035
360	453.476502	181.398315	217.8467407	316.0428772	0.999283373
384	503.3107	141.85936	217.0045471	304.4026489	0.999681473
408	518.854126	154.150238	220.354126	309.3813477	0.99978441
432	471.382324	222.746155	208.8736572	303.8798218	0.999748051
456	431.902161	174.890823	208.1036377	310.6296997	0.999846935

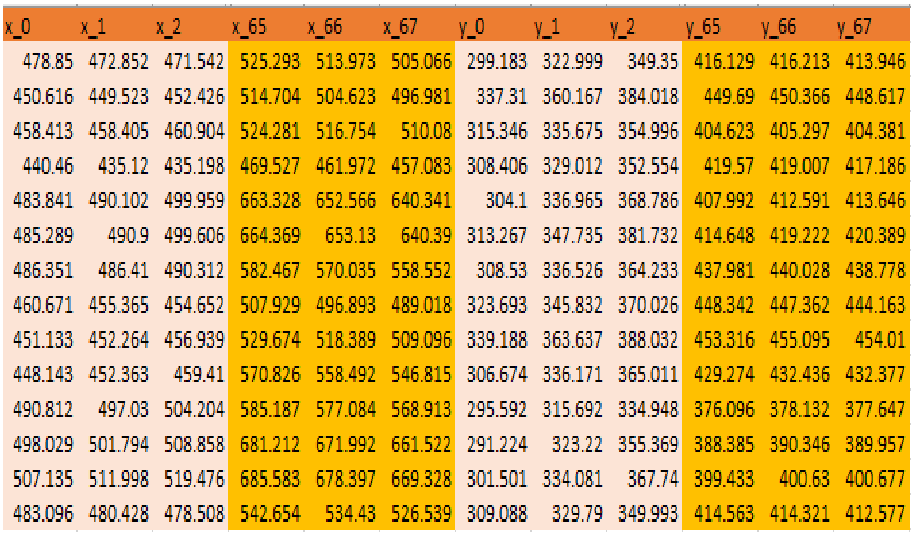
(a) (b)

Figure (4.11): The Face Detection Steps, (a) Represent Dimensions and Score of Faces, (b) Represent Face Detector

The second step is to find the Landmark, through which the action unit is calculated, as shown in Figure (4.12). After that, the action unit is found, where the number of frames that contain features is (14969). As for the number of action units, they are twenty types, as shown in the Table (4.15). In this table, ten frames of real video and ten frames of fake video are taken



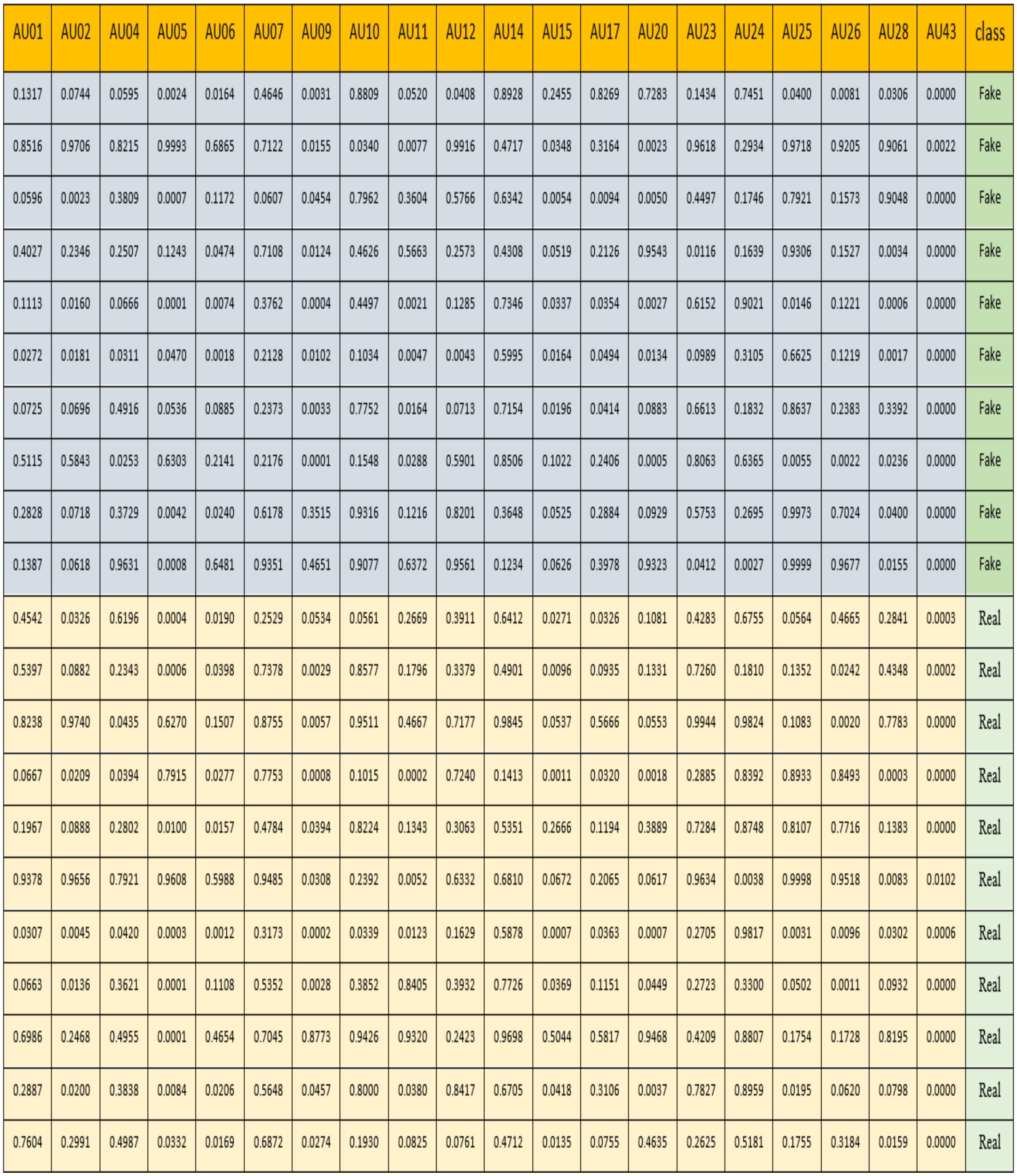
Location of the landmark



x_0	x_1	x_2	x_65	x_66	x_67	y_0	y_1	y_2	y_65	y_66	y_67
478.85	472.852	471.542	525.293	513.973	505.066	299.183	322.999	349.35	416.129	416.213	413.946
450.616	449.523	452.426	514.704	504.623	496.981	337.31	360.167	384.018	449.69	450.366	448.617
458.413	458.405	460.904	524.281	516.754	510.08	315.346	335.675	354.996	404.623	405.297	404.381
440.46	435.12	435.198	469.527	461.972	457.083	308.406	329.012	352.554	419.57	419.007	417.186
483.841	490.102	499.959	663.328	652.566	640.341	304.1	336.965	368.786	407.992	412.591	413.646
485.289	490.9	499.606	664.369	653.13	640.39	313.267	347.735	381.732	414.648	419.222	420.389
486.351	486.41	490.312	582.467	570.035	558.552	308.53	336.526	364.233	437.981	440.028	438.778
460.671	455.365	454.652	507.929	496.893	489.018	323.693	345.832	370.026	448.342	447.362	444.163
451.133	452.264	456.939	529.674	518.389	509.096	339.188	363.637	388.032	453.316	455.095	454.01
448.143	452.363	459.41	570.826	558.492	546.815	306.674	336.171	365.011	429.274	432.436	432.377
490.812	497.03	504.204	585.187	577.084	568.913	295.592	315.692	334.948	376.096	378.132	377.647
498.029	501.794	508.858	681.212	671.992	661.522	291.224	323.22	355.369	388.385	390.346	389.957
507.135	511.998	519.476	685.583	678.397	669.328	301.501	334.081	367.74	399.433	400.63	400.677
483.096	480.428	478.508	542.654	534.43	526.539	309.088	329.79	349.993	414.563	414.321	412.577

Figure (4.12): Landmark and the Values of a Landmark in each Face

Table (4.15): Profile of Extracted Features using Logistics Modeling of Barack Obama that is Extracted from each Frame



AU01	AU02	AU04	AU05	AU06	AU07	AU09	AU10	AU11	AU12	AU14	AU15	AU17	AU20	AU23	AU24	AU25	AU26	AU28	AU43	class
0.1317	0.0744	0.0595	0.0024	0.0164	0.4646	0.0031	0.8809	0.0520	0.0408	0.8928	0.2455	0.8269	0.7283	0.1434	0.7451	0.0400	0.0081	0.0306	0.0000	Fake
0.8516	0.9706	0.8215	0.9993	0.6865	0.7122	0.0155	0.0340	0.0077	0.9916	0.4717	0.0348	0.3164	0.0023	0.9618	0.2934	0.9718	0.9205	0.9061	0.0022	Fake
0.0596	0.0023	0.3809	0.0007	0.1172	0.0607	0.0454	0.7962	0.3604	0.5766	0.6342	0.0054	0.0094	0.0050	0.4497	0.1746	0.7921	0.1573	0.9048	0.0000	Fake
0.4027	0.2346	0.2507	0.1243	0.0474	0.7108	0.0124	0.4626	0.5663	0.2573	0.4308	0.0519	0.2126	0.9543	0.0116	0.1639	0.9306	0.1527	0.0034	0.0000	Fake
0.1113	0.0160	0.0666	0.0001	0.0074	0.3762	0.0004	0.4497	0.0021	0.1285	0.7346	0.0337	0.0354	0.0027	0.6152	0.9021	0.0146	0.1221	0.0006	0.0000	Fake
0.0272	0.0181	0.0311	0.0470	0.0018	0.2128	0.0102	0.1034	0.0047	0.0043	0.5995	0.0164	0.0494	0.0134	0.0989	0.3105	0.6625	0.1219	0.0017	0.0000	Fake
0.0725	0.0696	0.4916	0.0536	0.0885	0.2373	0.0033	0.7752	0.0164	0.0713	0.7154	0.0196	0.0414	0.0883	0.6613	0.1832	0.8637	0.2383	0.3392	0.0000	Fake
0.5115	0.5843	0.0253	0.6303	0.2141	0.2176	0.0001	0.1548	0.0288	0.5901	0.8506	0.1022	0.2406	0.0005	0.8063	0.6365	0.0055	0.0022	0.0236	0.0000	Fake
0.2828	0.0718	0.3729	0.0042	0.0240	0.6178	0.3515	0.9316	0.1216	0.8201	0.3648	0.0525	0.2884	0.0929	0.5753	0.2695	0.9973	0.7024	0.0400	0.0000	Fake
0.1387	0.0618	0.9631	0.0008	0.6481	0.9351	0.4651	0.9077	0.6372	0.9561	0.1234	0.0626	0.3978	0.9323	0.0412	0.0027	0.9999	0.9677	0.0155	0.0000	Fake
0.4542	0.0326	0.6196	0.0004	0.0190	0.2529	0.0534	0.0561	0.2669	0.3911	0.6412	0.0271	0.0326	0.1081	0.4283	0.6755	0.0564	0.4665	0.2841	0.0003	Real
0.5397	0.0882	0.2343	0.0006	0.0398	0.7378	0.0029	0.8577	0.1796	0.3379	0.4901	0.0096	0.0935	0.1331	0.7260	0.1810	0.1352	0.0242	0.4348	0.0002	Real
0.8238	0.9740	0.0435	0.6270	0.1507	0.8755	0.0057	0.9511	0.4667	0.7177	0.9845	0.0537	0.5666	0.0553	0.9944	0.9824	0.1083	0.0020	0.7783	0.0000	Real
0.0667	0.0209	0.0394	0.7915	0.0277	0.7753	0.0008	0.1015	0.0002	0.7240	0.1413	0.0011	0.0320	0.0018	0.2885	0.8392	0.8933	0.8493	0.0003	0.0000	Real
0.1967	0.0888	0.2802	0.0100	0.0157	0.4784	0.0394	0.8224	0.1343	0.3063	0.5351	0.2666	0.1194	0.3889	0.7284	0.8748	0.8107	0.7716	0.1383	0.0000	Real
0.9378	0.9656	0.7921	0.9608	0.5988	0.9485	0.0308	0.2392	0.0052	0.6332	0.6810	0.0672	0.2065	0.0617	0.9634	0.0038	0.9998	0.9518	0.0083	0.0102	Real
0.0307	0.0045	0.0420	0.0003	0.0012	0.3173	0.0002	0.0339	0.0123	0.1629	0.5878	0.0007	0.0363	0.0007	0.2705	0.9817	0.0031	0.0096	0.0302	0.0006	Real
0.0663	0.0136	0.3621	0.0001	0.1108	0.5352	0.0028	0.3852	0.8405	0.3932	0.7726	0.0369	0.1151	0.0449	0.2723	0.3300	0.0502	0.0011	0.0932	0.0000	Real
0.6986	0.2468	0.4955	0.0001	0.4654	0.7045	0.8773	0.9426	0.9320	0.2423	0.9698	0.5044	0.5817	0.9468	0.4209	0.8807	0.1754	0.1728	0.8195	0.0000	Real
0.2887	0.0200	0.3838	0.0084	0.0206	0.5648	0.0457	0.8000	0.0380	0.8417	0.6705	0.0418	0.3106	0.0037	0.7827	0.8959	0.0195	0.0620	0.0798	0.0000	Real
0.7604	0.2991	0.4987	0.0332	0.0169	0.6872	0.0274	0.1930	0.0825	0.0761	0.4712	0.0135	0.0755	0.4635	0.2625	0.5181	0.1755	0.3184	0.0159	0.0000	Real

- **Training Mode:**

The results of the model after the training are shown in Table (4.16).

Table (4.16): Accuracy and Loss Metrics in the Training the Model

k-fold=5	Accuracy	loss
Maximum	96.509%	0.11585%
Minimum	96.086%	0.10413%
Overall	96.222%	0.10986%

- **Testing Mode:**

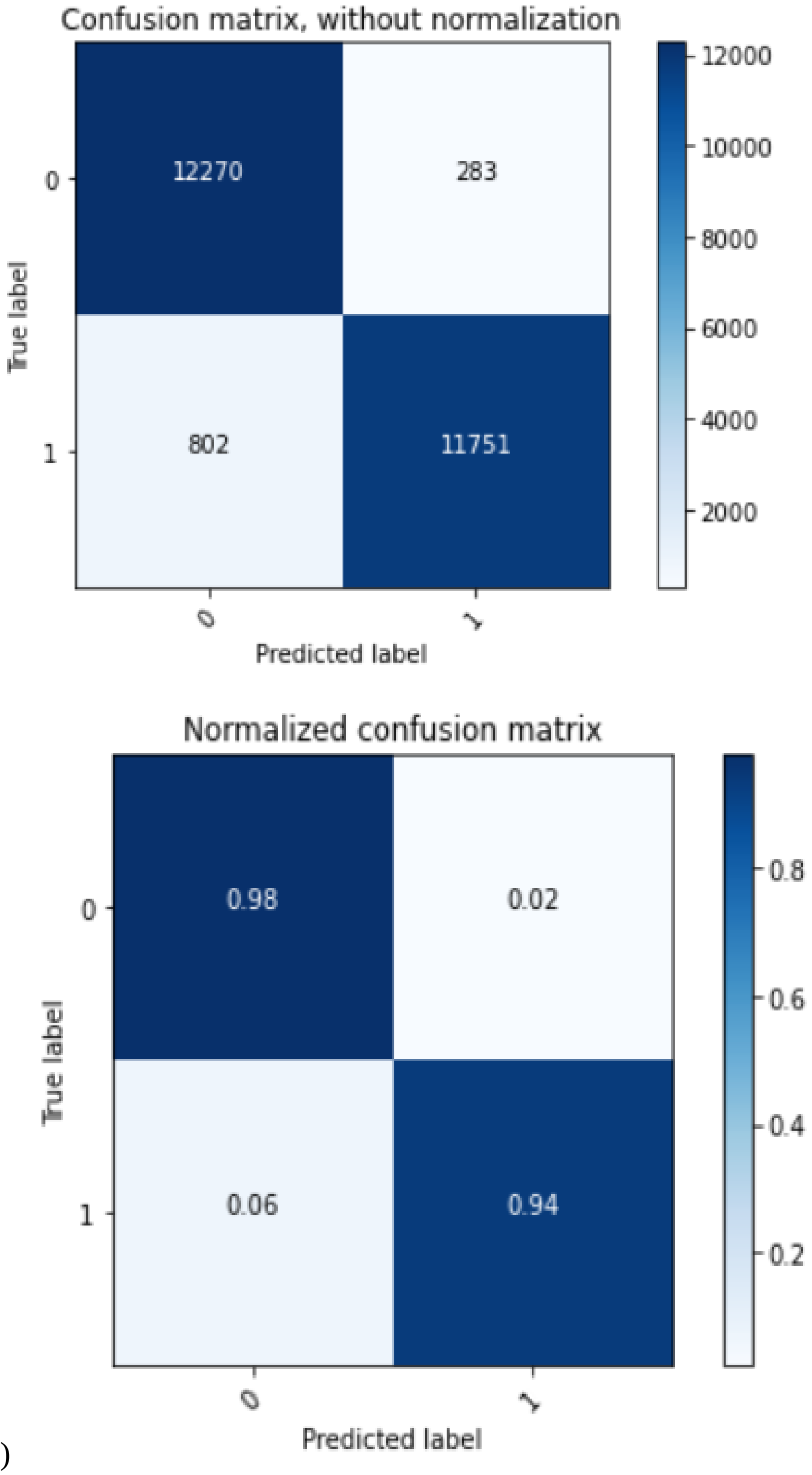
Table (4.17) shows that the measures of accuracy, recall, support, and F1 score are lower than those of the JAA-Net model. The model prediction is less compared to the profile that was extracted from the JAA-Net. The accuracy measures and the standard deviation of the general accuracy value of the model are also less than the JAA-Net model as shown in Table (4.18). The confusion matrix shows the prediction rates, and therefore the number of misclassification cases is greater compared to the JAA-Net model, as shown in Figure (4.13).

Table (4.17): Performance Metrics for the Model

k-fold	Type class	precision	Recall	F1-score	support	accuracy
K=5	Fake	98%	94%	96%	13072	96%
	Real	90%	95%	92%	12034	96%
average		96%	96%	96%		96%

Table (4.18): The Accuracy and Standard Deviation of the Accuracy Value

k-fold=5	value
Mean accuracy	95.678%
Standard deviation of accuracy	0.03891%



(a)

(b)

Figure (4.13): Confusion Matrix for the Profile Generated with Logistic, (a) The Confusion Matrix Without Normalized, (b) The Confusion Matrix with Normalized

4.7.4 BiLSTM Prediction with the Profile Created Using SVM

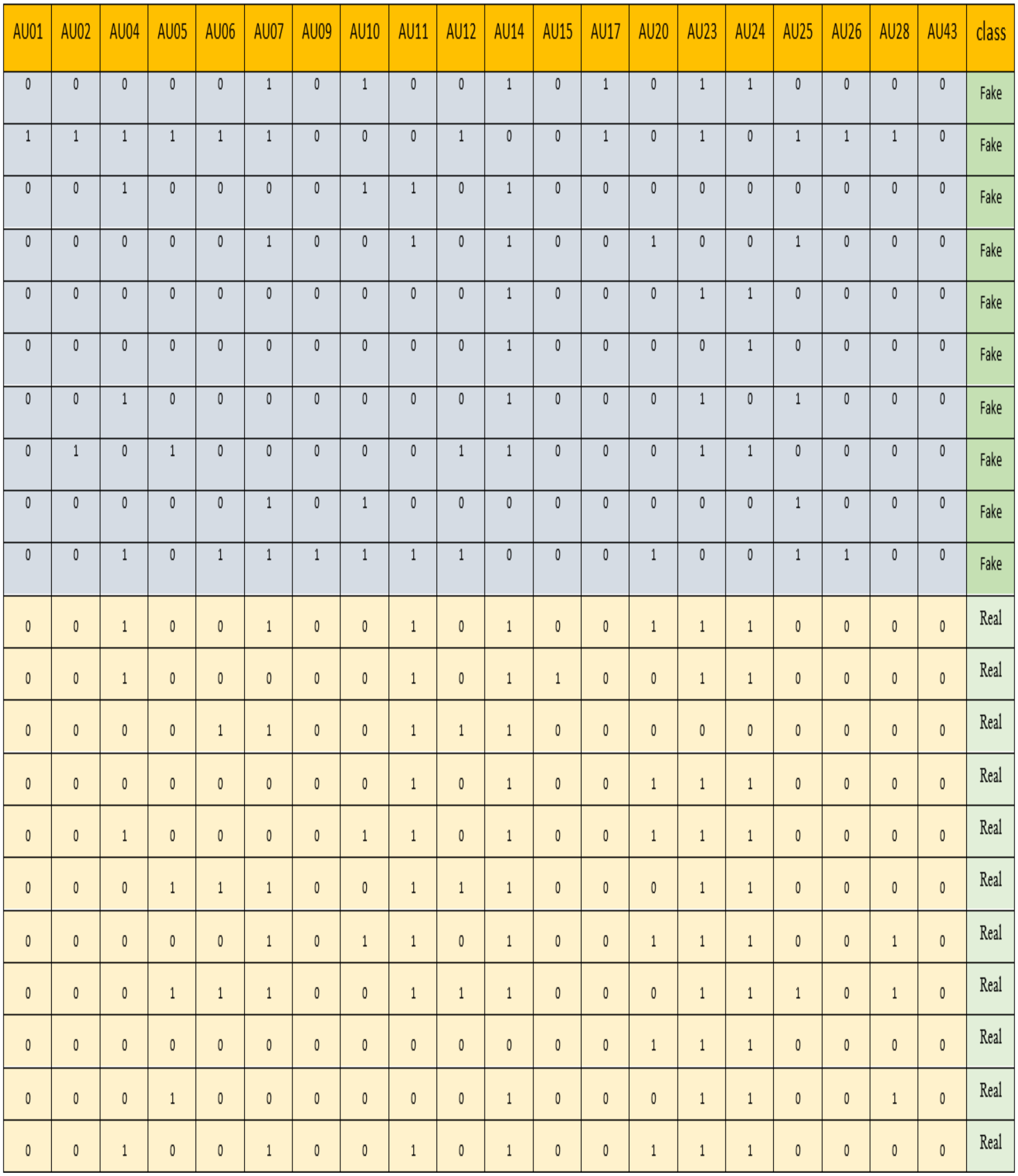
The steps used to create a profile using SVM are the same as those used in logistic modeling. Creating a profile using SVM modeling, depends on a set of steps. The first step is face detection, frames of videos that do not contain a face and are excluded. The second step is to find the Landmark, through which the action unit is calculated After that, the action unit is found, where the number of frames that contain features is (15960). As for the number of action units, there are twenty types, as shown in the Table (4.19). In this table, ten frames of real video and ten frames of fake video are taken.

The action unit values are either zero or one, and the reason for the model mechanism depends on the presence or absence of the action unit. The SVM model does not find the proportions of the existence of the action unit because the basis of its work is the binary classification. As a result of the action unit values being either zero or one, the model measures are low when compared with other models. Some action units between some people are close, so when the values of zero and one are calculated, the results are not satisfactory.

- **Training Mode:**

In the learning phase, it will be the same as the previous steps in terms of dividing the data. As well as the values of the epoch and the values of K-Fold terms, and then the prediction model. The results of the model after the training are shown in the Table (4.20).

Table (4.19): Profile of Extracted Features using SVM Modeling of Barack Obama



AU01	AU02	AU04	AU05	AU06	AU07	AU09	AU10	AU11	AU12	AU14	AU15	AU17	AU20	AU23	AU24	AU25	AU26	AU28	AU43	class
0	0	0	0	0	1	0	1	0	0	1	0	1	0	1	1	0	0	0	0	Fake
1	1	1	1	1	1	0	0	0	1	0	0	1	0	1	0	1	1	1	0	Fake
0	0	1	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	Fake
0	0	0	0	0	1	0	0	1	0	1	0	0	1	0	0	1	0	0	0	Fake
0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	Fake
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	Fake
0	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	Fake
0	1	0	1	0	0	0	0	0	1	1	0	0	0	1	1	0	0	0	0	Fake
0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	Fake
0	0	1	0	1	1	1	1	1	1	0	0	0	1	0	0	1	1	0	0	Fake
0	0	1	0	0	1	0	0	1	0	1	0	0	1	1	1	0	0	0	0	Real
0	0	1	0	0	0	0	0	1	0	1	1	0	0	1	1	0	0	0	0	Real
0	0	0	0	1	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	Real
0	0	0	0	0	0	0	0	1	0	1	0	0	1	1	1	0	0	0	0	Real
0	0	1	0	0	0	0	1	1	0	1	0	0	1	1	1	0	0	0	0	Real
0	0	0	1	1	1	0	0	1	1	1	0	0	0	1	1	0	0	0	0	Real
0	0	0	0	0	1	0	1	1	0	1	0	0	1	1	1	0	0	1	0	Real
0	0	0	1	1	1	0	0	1	1	1	0	0	0	1	1	1	0	1	0	Real
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	Real
0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	1	0	0	1	0	Real
0	0	1	0	0	1	0	0	1	0	1	0	0	1	1	1	0	0	0	0	Real

Table (4.20): Accuracy and Loss Metrics in the Training the Model

k-fold=5	Accuracy	loss
Maximum	83.318%	0.34229%
Minimum	83.059%	0.33788%
Overall	83.247%	0.33958%

- **Testing Mode:**

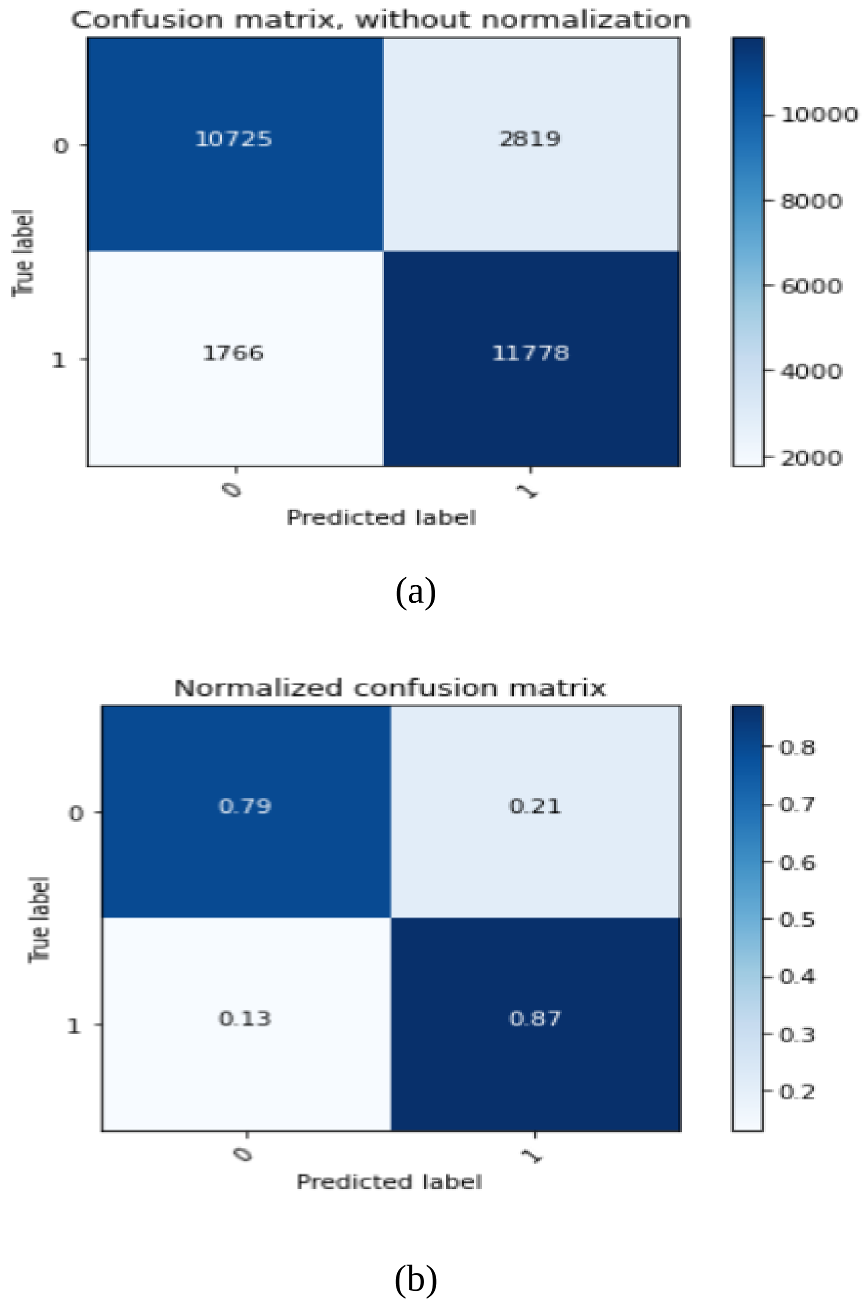
In the testing phase of the model, the same measures accuracy, recall, support, and F1 score are also used less than the measures of the JAA-Net and Logistics model, as shown in Table (4.21). The model predictability is lower compared to the profiles extracted from JAA-Net and Logistics. The accuracy measures and the standard deviation of the general accuracy value of the model are also less than the JAA-Net and Logistics models, as shown in Table (4.22). The confusion matrix shows the prediction rates, and therefore the number of misclassification cases is greater compared to the JAA-Net and Logistics models, as shown in Figure (4.14).

Table (4.21): Performance Metrics for the Model

k-fold	Type class	precision	Recall	F1-score	support	accuracy
K=5	Fake	79%	86%	82%	12491	83%
	Real	87%	81%	84%	14597	83%
average		83%	83%	83%		83%

Table (4.22): The Accuracy and Standard Deviation of the Accuracy Value

k-fold=5	value
Mean accuracy	83.073%
Standard deviation of accuracy	0.01465%



(a)

(b)

Figure (4.14): Confusion Matrix for the Profile Generated with SVM, (a) The confusion matrix without normalized, (b) The confusion matrix with normalized

4.7.5 BiLSTM Prediction with the Profile Created Using logistic and Face Pose

A profile used in the prediction contains features of the action unit and facial pose using two models, logistic and face pose. The profile contains twenty-three features, of which the logistic model extracts twenty and three by the face pose model. The number of frames containing features is 15960 using the two models, as shown in Table (4.23). In this Table, ten frames of real and ten frames of fake video are taken. Experiments with this profile in prediction models showed that the results are lower than those obtained from JAA-Net and face pose. The results show that the features extracted from this profile are less accurate than those extracted from JAA-Net and face pose together.

- **Training Mode:**

The same previous steps are used in the training phase. So are the epoch values and K-Fold terms values, and then the prediction model. The results of the model after training are shown in Table (4.24).

- **Testing Mode:**

In the testing phase, the measures are also less than those of the JAA-Net and pose model, as shown in Table (4.25). The model predictability is lower than the profiles extracted from JAA-Net and face pose. The accuracy measures and the standard deviation of the general accuracy value of the model are also less than the JAA-Net and face pose models, as shown in Table (4.26). The confusion matrix shows the prediction rates, and therefore, the number of misclassification cases is greater compared to the JAA-Net and face pose models, as shown in Figure (4.15).

Table (4.23): Profile of Extracted Features using Logistic and Face Pose Modeling of Barack Obama

AU01	AU02	AU04	AU05	AU06	AU07	AU09	AU10	AU11	AU12	AU14	AU15	AU17	AU20	AU23	AU24	AU25	AU26	AU28	AU43	Pitch	Roll	Yaw	class
0.0353	0.0811	0.7112	0.0000	0.0416	0.3635	0.4400	0.9141	0.7476	0.0019	0.0123	0.9972	0.7196	0.7097	0.8401	0.9187	0.9972	0.8323	0.0013	0.0004	17.4897	2.2490	32.6562	Fake
0.2722	0.0488	0.6165	0.0258	0.0415	0.5442	0.2465	0.2370	0.8905	0.0002	0.0008	0.9358	0.2563	0.0042	0.0905	0.2722	0.8122	0.0009	0.0004	0.0011	26.7008	2.6903	30.5619	Fake
0.0108	0.0075	0.2367	0.0011	0.5218	0.9636	0.9661	0.1263	0.0035	0.0404	0.0682	0.5054	0.8954	0.0684	0.1804	0.7545	0.9800	0.7826	0.0145	0.9956	17.1724	-7.2977	38.1214	Fake
0.0133	0.0894	0.0140	0.0075	0.3634	0.0502	0.0107	0.3324	0.0100	0.8360	0.4081	0.0271	0.6219	0.0281	0.8334	0.1765	0.8849	0.8361	0.1283	0.0000	6.4532	-27.6630	-21.6944	Fake
0.3903	0.7976	0.1731	0.0534	0.0659	0.3774	0.0068	0.8455	0.0001	0.0725	0.7731	0.3953	0.0070	0.4081	0.0340	0.2725	0.9872	0.8702	0.1365	0.0177	6.0617	-27.0322	-21.3952	Fake
0.8723	0.7605	0.5988	0.0062	0.0127	0.3687	0.0482	0.8217	0.0006	0.0855	0.5988	0.0699	0.0142	0.3051	0.0315	0.0631	0.9955	0.8501	0.1213	0.0001	5.2478	-27.0012	-21.6611	Fake
0.0535	0.0167	0.2952	0.9319	0.1015	0.5405	0.0012	0.7400	0.0002	0.8796	0.4959	0.2169	0.2632	0.0106	0.6931	0.1378	0.7711	0.3076	0.0142	0.0000	3.5378	-21.6460	-8.4324	Fake
0.0621	0.0875	0.0048	0.5248	0.0171	0.2426	0.0000	0.1991	0.0124	0.8914	0.5944	0.3470	0.8760	0.0367	0.6812	0.0782	0.9319	0.7722	0.0428	0.0000	1.5134	-11.0848	-3.6954	Fake
0.1081	0.1215	0.3533	0.0026	0.1793	0.6175	0.0347	0.1002	0.0176	0.0194	0.1254	0.0133	0.0088	0.0023	0.0456	0.1578	0.9716	0.5865	0.1376	0.4361	-3.9364	-10.7428	-1.6737	Fake
0.3636	0.0219	0.4669	0.0000	0.2953	0.3651	0.6434	0.9223	0.9662	0.5054	0.0529	0.0325	0.0526	0.3888	0.9640	0.0119	0.9996	0.8107	0.5341	0.0002	2.8306	-8.8937	-0.1617	Fake
0.9129	0.5840	0.4126	0.0016	0.0758	0.5250	0.1029	0.4446	0.5534	0.7655	0.9537	0.0583	0.0766	0.1096	0.6148	0.6891	0.0281	0.0293	0.0562	0.0237	-28.6488	-12.5088	-13.2660	Real
0.3516	0.1868	0.5118	0.0229	0.1465	0.7240	0.0138	0.8514	0.5206	0.5693	0.9818	0.0298	0.1489	0.0431	0.9005	0.9753	0.0847	0.0179	0.8493	0.0001	-28.7903	-14.1373	-9.5498	Real
0.3670	0.0714	0.8799	0.0041	0.1258	0.9338	0.7841	0.1149	0.2310	0.0375	0.8975	0.1376	0.5387	0.9272	0.8148	0.9948	0.0785	0.1206	0.7040	0.0000	-27.5732	-14.1260	-10.8419	Real
0.1050	0.1360	0.2927	0.0005	0.0177	0.0580	0.0047	0.0778	0.0569	0.1406	0.5591	0.0245	0.0803	0.1229	0.5088	0.9000	0.0337	0.0505	0.5282	0.0000	-24.8370	-14.2962	-10.6661	Real
0.1576	0.0253	0.0828	0.0004	0.1652	0.9186	0.0001	0.8973	0.3925	0.0606	0.7874	0.3760	0.5981	0.0198	0.6772	0.3510	0.0047	0.1064	0.1444	0.0000	-25.3415	-10.9825	-0.8076	Real
0.0187	0.0384	0.1837	0.0000	0.0801	0.2722	0.0574	0.7757	0.0295	0.0337	0.7186	0.5370	0.0895	0.1280	0.4163	0.7063	0.5669	0.3333	0.0455	0.0000	-24.8487	-10.1836	-3.4334	Real
0.1472	0.1347	0.4611	0.0001	0.0695	0.3050	0.0232	0.7133	0.1388	0.3591	0.5134	0.6191	0.3042	0.8068	0.2263	0.8070	0.4656	0.1742	0.0754	0.0000	-24.0031	-8.9285	-1.5519	Real
0.5779	0.9401	0.0155	0.9998	0.0103	0.8754	0.0619	0.2857	0.0850	0.0592	0.8274	0.0446	0.4205	0.4303	0.9193	0.8585	0.4400	0.1689	0.9544	0.0002	-6.9376	-5.6899	22.8188	Real
0.5934	0.0867	0.8941	0.0020	0.6966	0.7116	0.0809	0.9471	0.8637	0.2266	0.7547	0.3966	0.0832	0.0022	0.9193	0.6659	0.0354	0.0024	0.1716	0.0000	-1.6183	-5.8053	22.4540	Real
0.1154	0.0300	0.3388	0.0000	0.6097	0.6363	0.1377	0.6389	0.1845	0.3101	0.1678	0.3510	0.9656	0.0140	0.9158	0.5931	0.1224	0.2132	0.0091	0.0000	-3.9430	-5.5840	21.9353	Real
0.9129	0.5840	0.4126	0.0016	0.0758	0.5250	0.1029	0.4446	0.5534	0.7655	0.9537	0.0583	0.0766	0.1096	0.6148	0.6891	0.0281	0.0293	0.0562	0.0237	-28.6488	-12.5088	-13.2660	Real

Table (4.24): Accuracy and Loss Metrics in the Training the Model

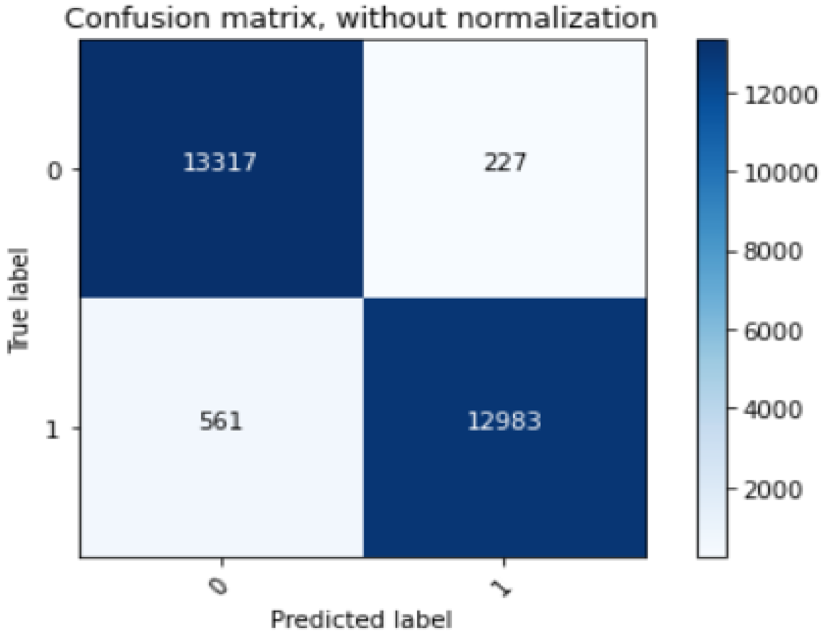
k-fold=5	Accuracy	loss
Maximum	97.651%	0.10225%
Minimum	96.806%	0.07784%
Overall	97.437%	0.08444%

Table (4.25): Performance Metrics for the Model

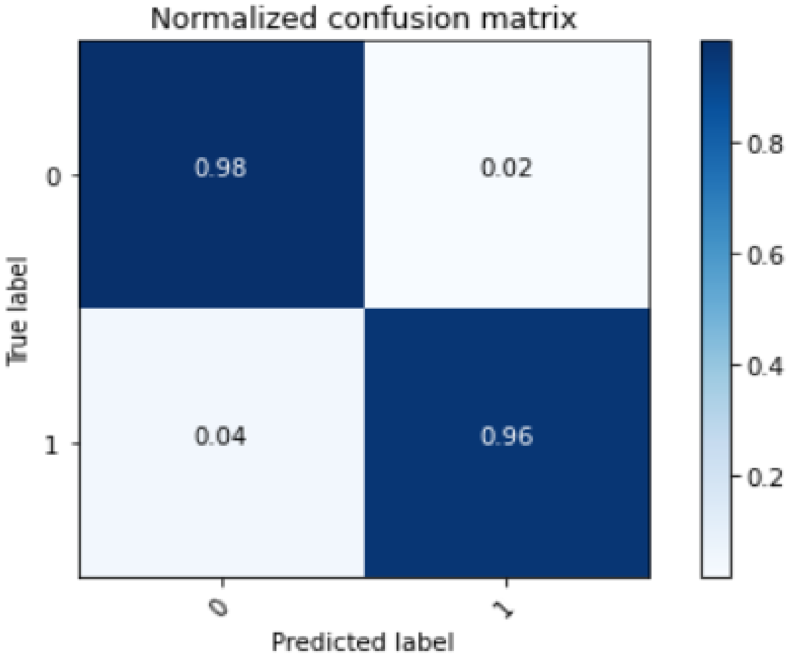
k-fold	Type class	precision	Recall	F1-score	support	accuracy
K=5	Fake(0)	98%	96%	97%	13878	97%
	Real(1)	96%	98%	97%	13210	97%
average		97%	97%	97%		97%

Table (4.26): The Accuracy and Standard Deviation of the Accuracy Value

k-fold=5	value
Mean accuracy	97.091%
Standard deviation of accuracy	0.03112%



(a)



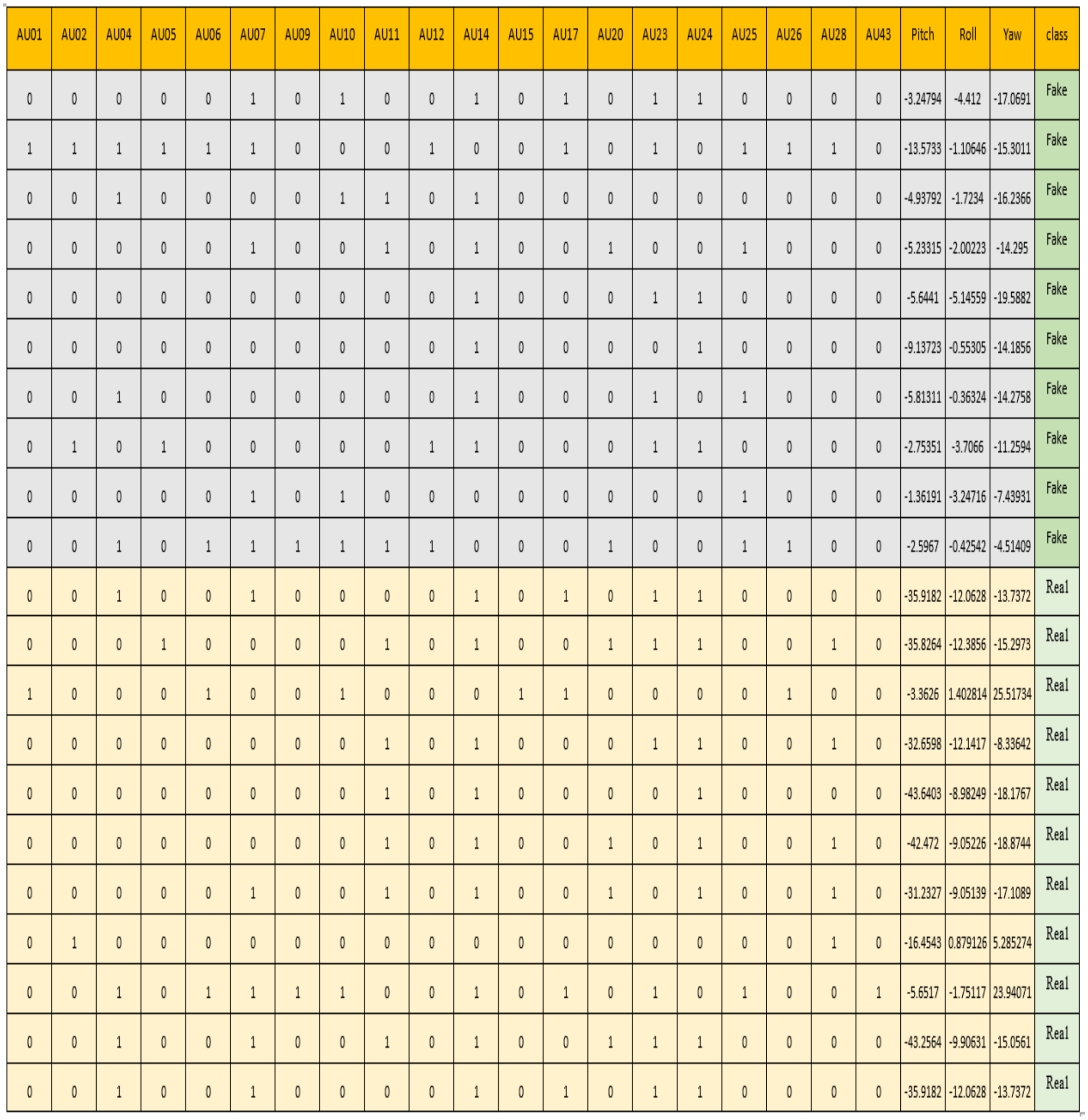
(b)

Figure (4.15): Confusion Matrix for the Profile Generated with logistic and Face Pose, (a) The Confusion Matrix without Normalized, (b) The Confusion Matrix with Normalized

4.7.6 BiLSTM Prediction with the Profile Created Using (SVM and Face Pose)

A profile contains features of the action unit and facial pose using two models, SVM and face pose. The profile contains Twenty-three features, of which twenty are extracted by the SVM model and three by the face pose model. The number of frames containing features is (14409) using the two models as shown in Table (4.27). In this Table, ten frames of real video and ten frames of fake video are taken for example. Experiments with this profile in prediction models showed that the results are lower than the results obtained from (JAA-Net and face pose together) and lower (Logistics and face pose together). The lower results obtained show that the features extracted from this profile are less accurate compared to (JAA-Net and face pose together) and (Logistics and face pose together).

Table (4.27): Profile of Extracted Features using SVM and Face Pose Modeling of Barack Obama that are Extracted from each Frame



AU01	AU02	AU04	AU05	AU06	AU07	AU09	AU10	AU11	AU12	AU14	AU15	AU17	AU20	AU23	AU24	AU25	AU26	AU28	AU43	Pitch	Roll	Yaw	class
0	0	0	0	0	1	0	1	0	0	1	0	1	0	1	1	0	0	0	0	-3.24794	-4.412	-17.0691	Fake
1	1	1	1	1	1	0	0	0	1	0	0	1	0	1	0	1	1	1	0	-13.5733	-1.10646	-15.3011	Fake
0	0	1	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	-4.93792	-1.7234	-16.2366	Fake
0	0	0	0	0	1	0	0	1	0	1	0	0	1	0	0	1	0	0	0	-5.23315	-2.00223	-14.295	Fake
0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	-5.6441	-5.14559	-19.5882	Fake
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	-9.13723	-0.55305	-14.1856	Fake
0	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	-5.81311	-0.36324	-14.2758	Fake
0	1	0	1	0	0	0	0	0	1	1	0	0	0	1	1	0	0	0	0	-2.75351	-3.7066	-11.2594	Fake
0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	-1.36191	-3.24716	-7.43931	Fake
0	0	1	0	1	1	1	1	1	1	0	0	0	1	0	0	1	1	0	0	-2.5967	-0.42542	-4.51409	Fake
0	0	1	0	0	1	0	0	0	0	1	0	1	0	1	1	0	0	0	0	-35.9182	-12.0628	-13.7372	Real
0	0	0	1	0	0	0	0	1	0	1	0	0	1	1	1	0	0	1	0	-35.8264	-12.3856	-15.2973	Real
1	0	0	0	1	0	0	1	0	0	0	1	1	0	0	0	0	1	0	0	-3.3626	1.402814	25.51734	Real
0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	1	0	0	1	0	-32.6598	-12.1417	-8.33642	Real
0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	-43.6403	-8.98249	-18.1767	Real
0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1	0	0	1	0	-42.472	-9.05226	-18.8744	Real
0	0	0	0	0	1	0	0	1	0	1	0	0	1	0	1	0	0	1	0	-31.2327	-9.05139	-17.1089	Real
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	-16.4543	0.879126	5.285274	Real
0	0	1	0	1	1	1	1	0	0	1	0	1	0	1	0	1	0	0	1	-5.6517	-1.75117	23.94071	Real
0	0	1	0	0	1	0	0	1	0	1	0	0	1	1	1	0	0	0	0	-43.2564	-9.90631	-15.0561	Real
0	0	1	0	0	1	0	0	0	0	1	0	1	0	1	1	0	0	0	0	-35.9182	-12.0628	-13.7372	Real

- **Training Mode:**

The results of training in the prediction model showed that the accuracy is less than the accuracy of the prediction using two models (JAA-Net & face pose), as shown in the Table (4.28).

Table (4.28): Accuracy and Loss Metrics in the Training the Model

k-fold=5	Accuracy	loss
Maximum	84.934%	0.38675%
Minimum	82.245%	0.32791%
Overall	84.209%	0.34217%

- **Testing Mode:**

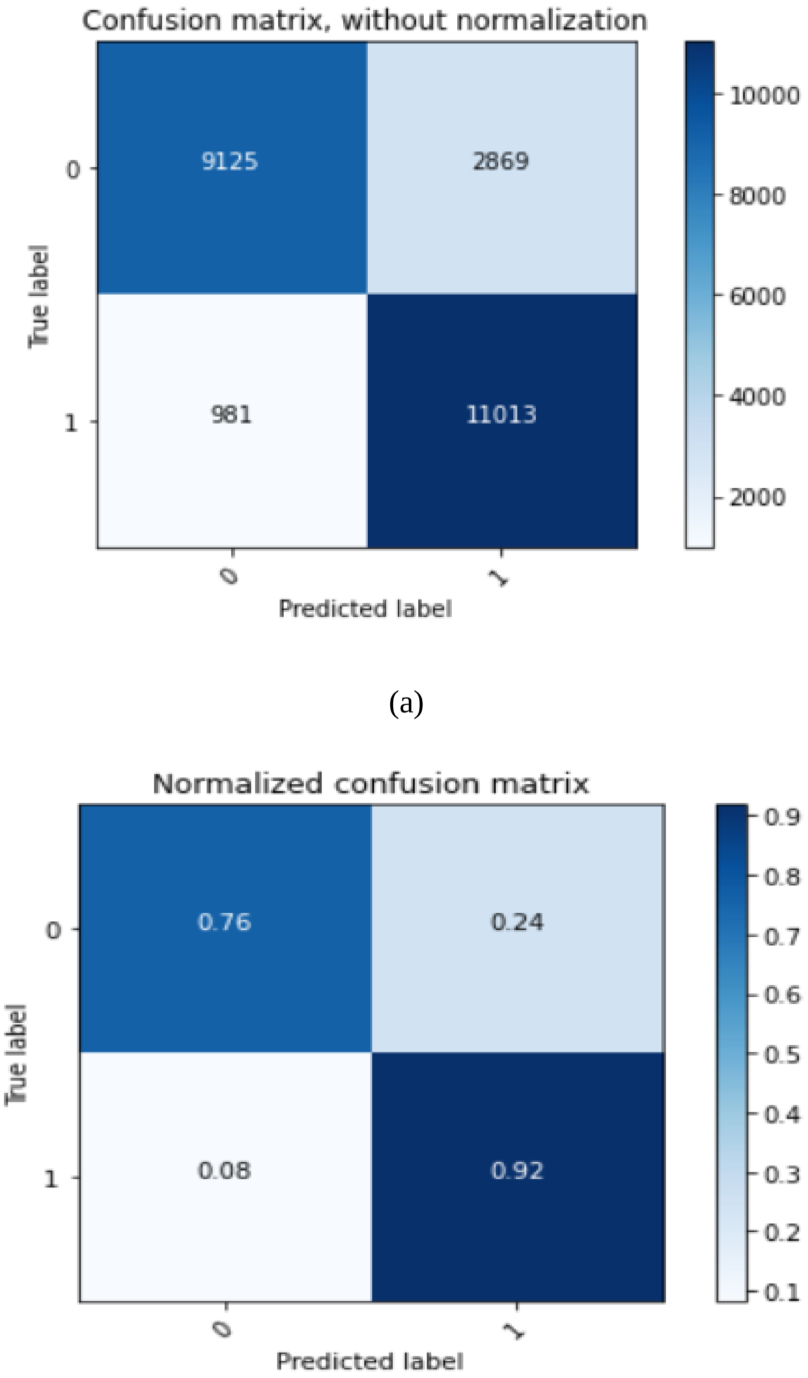
In the model testing phase, the measures used in the previous methods are also used and less than those of the JAA-Net & face pose model, as shown in Table (4.29). The accuracy measures and standard deviation of the method's general accuracy value are also lower than the JAA-Net & face pose models, as shown in Table (4.30). The confusion matrix shows the prediction rates, and therefore, the number of misclassification cases is greater than the JAA-Net & face pose models, as shown in Figure (4.16).

Table (4.29): Performance Metrics for the Model

k-fold	Type class	precision	Recall	F1-score	support	accuracy
K=5	Fake	76%	90%	83%	10106	84%
	Real	92%	79%	85%	13882	84%
average		84%	84%	84%		84%

Table (4.30): The Accuracy and Standard Deviation of the Accuracy Value

k-fold=5	value
Mean accuracy	83.950%
Standard deviation of accuracy	0.01098%



(a)

(b)

Figure (4.16): Confusion Matrix for the Profile Generated with SVM and face pose, (a) The Confusion Matrix Without Normalized, (b) The Confusion Matrix with Normalized

4.7.7 Summary Comparison Results of BiLSTM with Models of Feature Extraction

The results obtained through the experiments in the previous steps in dataset of Barak Obama show that the JAA-Net and face pose models are the best among all the results, as shown in the Table (4.31).

Table (4.31): Summary Comparison Results of all Models

	SVM	**logistic**	**JAA-Net**	**Face Pose**	**SVM + Pose**	**Logistic + pose**	**JAA-Net + pose**
No. Frame	15960	14969	14381	14426	14409	15960	14426
No. action unit	20	20	12	0	20	20	12
No. pose	0	0	0	3	3	3	3
Accuracy in Training	83.247%	96.222%	97.102 %	81.358%	84.209%	97.437%	**99.521%**
Accuracy in Testing	83.073%	95.678%	96.931%	81.482%	83.950%	97.091%	**99.403%**

4.8 Comparison Results of BiLSTM with the Other Dataset

Features are extracted from each frame of a Forensics++ dataset using two models (JAA-Net & face pose model), and the features are stored in a profile. This profile contains all the extracted features (action unit and face pose). The number of features is Fifteen, in addition to the label shown in Table (4.32). When features are extracted from real videos, the label will be (1), and for fake videos, the label will be (0). The profile extracted from the Forensics++ dataset, which contains real and fake data, is entered into the BiLSTM network for training.

Table (4.32): Profile of Extracted Features of Forensics++ that are Extracted from each Frame

AU01	AU02	AU04	AU06	AU07	AU10	AU12	AU14	AU15	AU17	AU23	AU24	Pitch	Roll	Yaw	class
0.019545304	0.30177268	0.06708989	0.0646278	0.9143954	0.006182177	0.7624282	0.09405329	0.7848908	0.000140231	0.001222117	7.91E-07	1.840897415	4.432400136	0.235413344	**1**
0.01179825	0.17291817	0.14307198	0.08013242	0.9478338	0.006669165	0.7046588	0.11960984	0.89986753	0.00018981	0.001389409	6.93E-07	1.736435326	4.113003023	0.810573782	**1**
0.012263583	0.17641237	0.13301255	0.06623663	0.9463905	0.006147038	0.6770891	0.1145826	0.8878221	0.000197846	0.001566301	7.89E-07	1.112825007	4.364332246	2.416476551	**1**
0.013063178	0.26620787	0.23680441	0.010492745	0.8557546	0.000741897	0.22215447	0.20434326	0.70382243	0.000274795	0.00086426	9.02E-07	1.021793963	4.325578946	2.182365858	**1**
0.021635665	0.22402547	0.27790165	0.010386084	0.8415666	0.000990682	0.24167252	0.22521497	0.677165	0.000361249	0.000899528	1.38E-06	0.940839714	4.133453595	2.027663093	**1**
0.022226635	0.21092175	0.27756363	0.025582016	0.9359161	0.001739307	0.43395093	0.18771027	0.8409399	0.000350938	0.001315985	1.52E-06	0.997947849	4.785117745	1.799730718	**1**
0.024628969	0.26935115	0.19539471	0.09792946	0.96605116	0.011505934	0.69358677	0.07375867	0.8887732	0.000162965	0.000859671	4.11E-07	0.733265759	5.516650655	1.633310763	**1**
0.02407511	0.29401404	0.20242351	0.11104371	0.9673244	0.011368264	0.6937599	0.06397989	0.89525443	0.000176663	0.000828534	4.25E-07	1.164007183	5.722698732	1.737976598	**1**
0.017939145	0.3529987	0.1246707	0.07125767	0.9353643	0.009480684	0.6905526	0.086033285	0.8918262	0.000156753	0.000828726	6.81E-07	1.105880497	5.681716397	2.918490333	**1**
0.027720522	0.37692982	0.14617155	0.086052276	0.9465839	0.006403605	0.6080587	0.09866961	0.8455045	0.00014664	0.000936338	5.53E-07	1.128688518	5.503234211	4.259410674	**1**
0.034825	0.670145	0.025244	0.711489	0.972587	0.019646	0.919511	0.025015	0.802506	0.000099	0.002096	0.000000	4.972511	-2.023465	4.735337	**0**
0.034825	0.670145	0.025244	0.711489	0.972587	0.019646	0.919511	0.025015	0.802506	0.000099	0.002096	0.000000	5.000736	-2.018120	4.714308	**0**
0.036735	0.686136	0.024819	0.711636	0.972952	0.019439	0.920819	0.026719	0.811943	0.000105	0.002195	0.000000	5.017653	-2.065144	4.732344	**0**
0.041514	0.648196	0.028231	0.678409	0.970930	0.017244	0.918698	0.028133	0.799157	0.000100	0.002232	0.000000	5.018407	-2.082907	4.691861	**0**
0.042652	0.655646	0.028636	0.677029	0.971660	0.017078	0.917282	0.028179	0.799952	0.000103	0.002228	0.000000	5.044405	-2.000767	4.626404	**0**
0.044595	0.655145	0.033399	0.667280	0.974019	0.015347	0.906204	0.026743	0.779036	0.000099	0.001911	0.000000	5.020577	-2.049118	4.613914	**0**
0.036014	0.687099	0.027937	0.700014	0.976128	0.017088	0.908060	0.023846	0.784487	0.000098	0.001963	0.000000	5.126463	-2.025143	4.707792	**0**
0.034277	0.674772	0.028735	0.695758	0.974138	0.018535	0.916094	0.025668	0.787152	0.000101	0.002324	0.000000	5.038454	-2.067233	4.712346	**0**
0.034151	0.676520	0.028133	0.695934	0.972971	0.018568	0.915072	0.026641	0.795033	0.000101	0.002333	0.000000	4.893482	-1.977711	4.663659	**0**
0.033794	0.668260	0.025911	0.672779	0.969678	0.016710	0.911282	0.027022	0.783920	0.000093	0.002120	0.000000	5.106585	-1.379855	4.625640	**0**

- **Training Mode:**

The same previous steps are used in the learning phase in terms of data split, epoch values, and K-Fold term values. The results of the model after training are shown in Table (4.33).

Table (4.33): Accuracy and Loss Metrics in the Training the Model

k-fold=5	Accuracy	loss
Maximum	95.981%	0.69224%
Minimum	69.321%	0.13030%
Overall	90.460%	0.24866%

- **Testing Mode:**

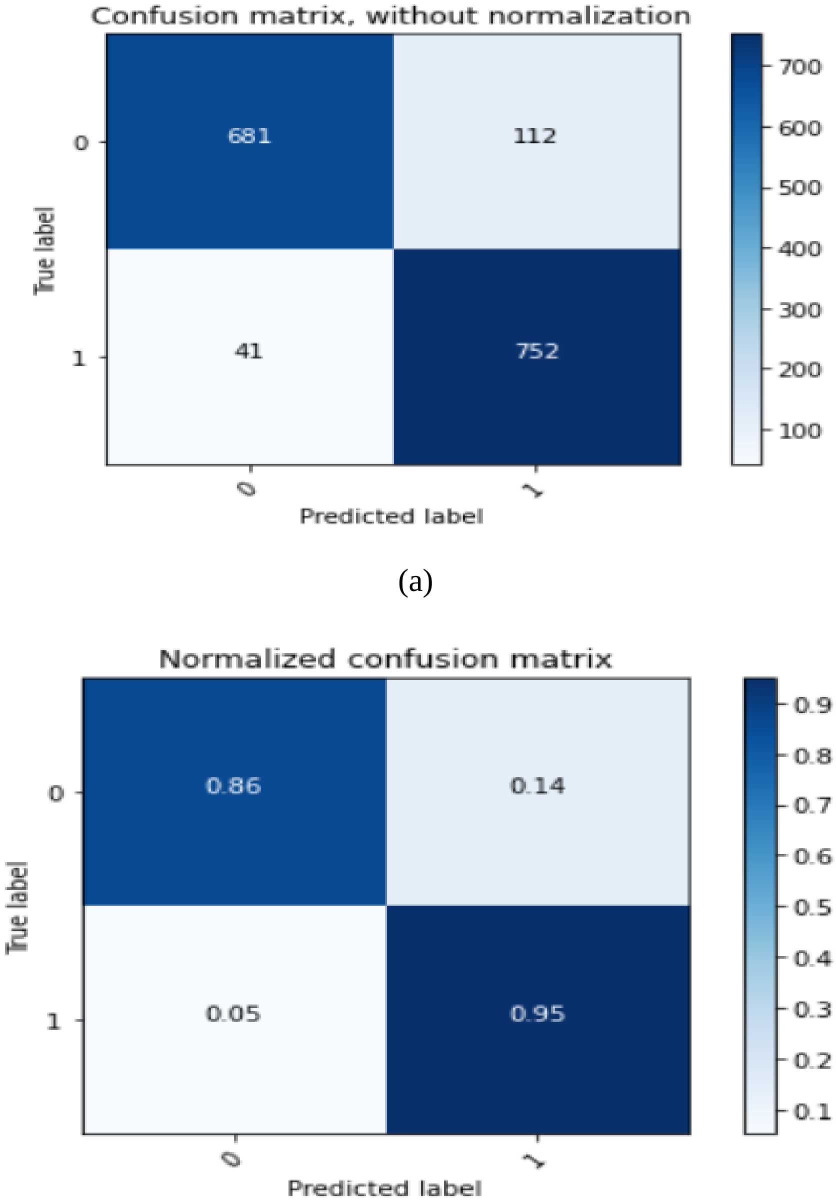
In the testing phase, results of the accuracy, recall, support, and F1 score appear, as shown in Table (4.34). The model predictability is lower than the profiles extracted from the dataset Barak Obama. The accuracy measures and the standard deviation of the general accuracy value of the model are also less than the dataset Barak Obama, as shown in Table (4.35). The confusion matrix shows the prediction rates; therefore, the misclassification cases are greater than Barak Obama's dataset, as shown in Figure (4.17).

Table (4.34): Performance Metrics for the Model

k-fold	Type class	precision	Recall	F1-score	support	accuracy
K=5	Fake	86%	94%	90%	722	90%
	Real	95%	87%	91%	864	90%
average		90%	90%	90%		90%

Table (4.35): The Accuracy and Standard Deviation of the Accuracy Value

k-fold=5	value
Mean accuracy	90367%
Standard deviation of accuracy	0.11718%



(a)

(b)

Figure (4.17): Confusion Matrix for the Profile Generated with the other dataset, (a) The Confusion Matrix Without Normalized, (b) The Confusion Matrix with Normalized

4.9 Comparison of Model's Prediction Using the Barack Obama Dataset

Many prediction models are used, as shown in the Table (4.36). The models (ANN, LSTM, GRU, and BiLSTM) are chosen because they give good results. The best model is BiLSTM regarding accuracy and the ability to predict fake and real videos. These models are tested using the Barack Obama dataset.

Table (4.36): Comparison of Prediction Models

Model	**JAA-Net**	**Face Pose**	**JAA-Net** **+** **face pose**
ANN	81.01%	91.28%	93.46%
LSTM	87.531%	83.40%	92.19%
GRU	87.56%	83.40%	87.15%
BiLSTM	96.931%	81.482%	**99.403%**

4.10 Comparison between the Results of the Traditional Methods and the Proposed System

There is very little research on behavioral methods for detecting deep fakes. Some research relies on forensic medicine, and other researchers follow eye movement to detect the difference. Also, some research uses changing human color or appearance, such as the Meso model. There is one study that used action unit, but only for real videos, and the classification method is SVM. The accuracy of the research and the proposed system is shown in the Table (4.37).

Table (4.37): Comparison of The Traditional Methods and The Proposed System

No. of works	Processing method	Feature Types	Dataset	Best results (Accuracy)
[31]	SVM classifier	3D head poses	UADFV	89%
[32]	Meso-4, MesoInception-4	Mesoscopic properties	Face2Face dataset, Deepfake dataset	98% for Deepfake 95% for Face2Face
[33]	one-class support vector machine	Facial action unit and face pose	Barack Obama	94%
[19]	LSTM	upper body language analysis	Barack Obama	94.39%
[36]	SVM	Facial Landmark and head pose	Barack Obama	89.6%
[37]	YOLO-CRNNs, Bi-LSTM	face regions	Celeb-DF, Forensics++	89.38%
[38]	SVM	artifact of facial (gestural, head stance, eyes)	World leaders, Forensics++	From 82.80% To 95.21%
The proposed system	LSTM	Action unit & face pose	Barack Obama	92.19%
The proposed system	BILSTM	Action unit & face pose	Barack Obama	99.403%

4.11 A Case Study of Unseen Video Prediction Experiments

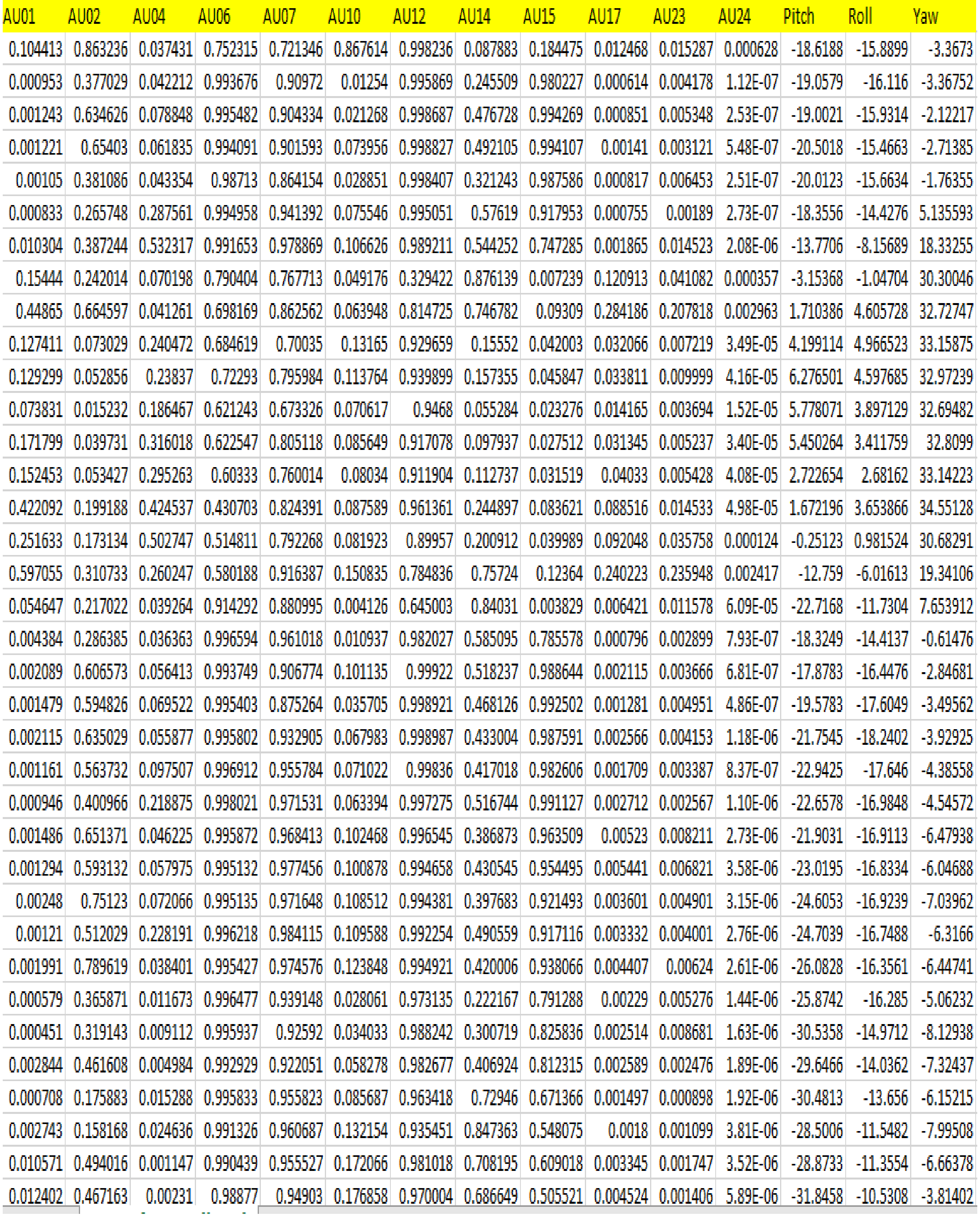
The prediction model is tested on unseen videos and not in the datasets. These videos of Barack Obama are collected from YouTube. The purpose of these new videos, which do not exist in the datasets, is to test the model's efficiency in predicting real and fake videos. The prediction process in the model takes place in two stages. The first stage is to extract features from the video using a model (JAA-Net and hybrid face pose). The second stage is prediction, where the model reads each frame and determines whether it is real or fake. The model uses address (1) for the real frame and address (0) for the fake frame.

4.11.1 The Real Video Example Unseen in the Model

The real video used in the method is downloaded from YouTube and lasts two minutes, as shown in Figure (4.18). There are 637 records containing features, as shown in Table (4.38). Each record is a vector of facial expression features (action unit and facial pose) extracted from each frame. Any frame containing facial expressions with behavior similar to real facial expressions is determined as real; Otherwise, it is fake.

The model results show that the number of tires detected as fake is zero. The number of frames detected as real is 637. The percentage of this real video is 100% because it contains real features (behavior) belonging to Obama, as shown in Table (4.39).

Table (4.38): The Extracted Features from The Real Video of Barack Obama



AU01	AU02	AU04	AU06	AU07	AU10	AU12	AU14	AU15	AU17	AU23	AU24	Pitch	Roll	Yaw
0.104413	0.863236	0.037431	0.752315	0.721346	0.867614	0.998236	0.087883	0.184475	0.012468	0.015287	0.000628	-18.6188	-15.8899	-3.3673
0.000953	0.377029	0.042212	0.993676	0.90972	0.01254	0.995869	0.245509	0.980227	0.000614	0.004178	1.12E-07	-19.0579	-16.116	-3.36752
0.001243	0.634626	0.078848	0.995482	0.904334	0.021268	0.998687	0.476728	0.994269	0.000851	0.005348	2.53E-07	-19.0021	-15.9314	-2.12217
0.001221	0.65403	0.061835	0.994091	0.901593	0.073956	0.998827	0.492105	0.994107	0.00141	0.003121	5.48E-07	-20.5018	-15.4663	-2.71385
0.00105	0.381086	0.043354	0.98713	0.864154	0.028851	0.998407	0.321243	0.987586	0.000817	0.006453	2.51E-07	-20.0123	-15.6634	-1.76355
0.000833	0.265748	0.287561	0.994958	0.941392	0.075546	0.995051	0.57619	0.917953	0.000755	0.00189	2.73E-07	-18.3556	-14.4276	5.135593
0.010304	0.387244	0.532317	0.991653	0.978869	0.106626	0.989211	0.544252	0.747285	0.001865	0.014523	2.08E-06	-13.7706	-8.15689	18.33255
0.15444	0.242014	0.070198	0.790404	0.767713	0.049176	0.329422	0.876139	0.007239	0.120913	0.041082	0.000357	-3.15368	-1.04704	30.30046
0.44865	0.664597	0.041261	0.698169	0.862562	0.063948	0.814725	0.746782	0.09309	0.284186	0.207818	0.002963	1.710386	4.605728	32.72747
0.127411	0.073029	0.240472	0.684619	0.70035	0.13165	0.929659	0.15552	0.042003	0.032066	0.007219	3.49E-05	4.199114	4.966523	33.15875
0.129299	0.052856	0.23837	0.72293	0.795984	0.113764	0.939899	0.157355	0.045847	0.033811	0.009999	4.16E-05	6.276501	4.597685	32.97239
0.073831	0.015232	0.186467	0.621243	0.673326	0.070617	0.9468	0.055284	0.023276	0.014165	0.003694	1.52E-05	5.778071	3.897129	32.69482
0.171799	0.039731	0.316018	0.622547	0.805118	0.085649	0.917078	0.097937	0.027512	0.031345	0.005237	3.40E-05	5.450264	3.411759	32.8099
0.152453	0.053427	0.295263	0.60333	0.760014	0.08034	0.911904	0.112737	0.031519	0.04033	0.005428	4.08E-05	2.722654	2.68162	33.14223
0.422092	0.199188	0.424537	0.430703	0.824391	0.087589	0.961361	0.244897	0.083621	0.088516	0.014533	4.98E-05	1.672196	3.653866	34.55128
0.251633	0.173134	0.502747	0.514811	0.792268	0.081923	0.89957	0.200912	0.039989	0.092048	0.035758	0.000124	-0.25123	0.981524	30.68291
0.597055	0.310733	0.260247	0.580188	0.916387	0.150835	0.784836	0.75724	0.12364	0.240223	0.235948	0.002417	-12.759	-6.01613	19.34106
0.054647	0.217022	0.039264	0.914292	0.880995	0.004126	0.645003	0.84031	0.003829	0.006421	0.011578	6.09E-05	-22.7168	-11.7304	7.653912
0.004384	0.286385	0.036363	0.996594	0.961018	0.010937	0.982027	0.585095	0.785578	0.000796	0.002899	7.93E-07	-18.3249	-14.4137	-0.61476
0.002089	0.606573	0.056413	0.993749	0.906774	0.101135	0.99922	0.518237	0.988644	0.002115	0.003666	6.81E-07	-17.8783	-16.4476	-2.84681
0.001479	0.594826	0.069522	0.995403	0.875264	0.035705	0.998921	0.468126	0.992502	0.001281	0.004951	4.86E-07	-19.5783	-17.6049	-3.49562
0.002115	0.635029	0.055877	0.995802	0.932905	0.067983	0.998987	0.433004	0.987591	0.002566	0.004153	1.18E-06	-21.7545	-18.2402	-3.92925
0.001161	0.563732	0.097507	0.996912	0.955784	0.071022	0.99836	0.417018	0.982606	0.001709	0.003387	8.37E-07	-22.9425	-17.646	-4.38558
0.000946	0.400966	0.218875	0.998021	0.971531	0.063394	0.997275	0.516744	0.991127	0.002712	0.002567	1.10E-06	-22.6578	-16.9848	-4.54572
0.001486	0.651371	0.046225	0.995872	0.968413	0.102468	0.996545	0.386873	0.963509	0.00523	0.008211	2.73E-06	-21.9031	-16.9113	-6.47938
0.001294	0.593132	0.057975	0.995132	0.977456	0.100878	0.994658	0.430545	0.954495	0.005441	0.006821	3.58E-06	-23.0195	-16.8334	-6.04688
0.00248	0.75123	0.072066	0.995135	0.971648	0.108512	0.994381	0.397683	0.921493	0.003601	0.004901	3.15E-06	-24.6053	-16.9239	-7.03962
0.00121	0.512029	0.228191	0.996218	0.984115	0.109588	0.992254	0.490559	0.917116	0.003332	0.004001	2.76E-06	-24.7039	-16.7488	-6.3166
0.001991	0.789619	0.038401	0.995427	0.974576	0.123848	0.994921	0.420006	0.938066	0.004407	0.00624	2.61E-06	-26.0828	-16.3561	-6.44741
0.000579	0.365871	0.011673	0.996477	0.939148	0.028061	0.973135	0.222167	0.791288	0.00229	0.005276	1.44E-06	-25.8742	-16.285	-5.06232
0.000451	0.319143	0.009112	0.995937	0.92592	0.034033	0.988242	0.300719	0.825836	0.002514	0.008681	1.63E-06	-30.5358	-14.9712	-8.12938
0.002844	0.461608	0.004984	0.992929	0.922051	0.058278	0.982677	0.406924	0.812315	0.002589	0.002476	1.89E-06	-29.6466	-14.0362	-7.32437
0.000708	0.175883	0.015288	0.995833	0.955823	0.085687	0.963418	0.72946	0.671366	0.001497	0.000898	1.92E-06	-30.4813	-13.656	-6.15215
0.002743	0.158168	0.024636	0.991326	0.960687	0.132154	0.935451	0.847363	0.548075	0.0018	0.001099	3.81E-06	-28.5006	-11.5482	-7.99508
0.010571	0.494016	0.001147	0.990439	0.955527	0.172066	0.981018	0.708195	0.609018	0.003345	0.001747	3.52E-06	-28.8733	-11.3554	-6.66378
0.012402	0.467163	0.00231	0.98877	0.94903	0.176858	0.970004	0.686649	0.505521	0.004524	0.001406	5.89E-06	-31.8458	-10.5308	-3.81402

Figure (4.18): The Real Video of Barack Obama

Table (4.39): Results of the Real Video Example

Count of frames fake	0
Count of frames real	637
Percentage of this video is fake	0.0%
Percentage of this video is real	100.0%

4.11.2 The Fake Video Example Unseen in the Model

The fake video used in the model is downloaded from YouTube and is 33 seconds long, as shown in Figure (4.19). There are 318 records containing features, as shown in Table (4.40). In this Table, twenty frames of fake video are taken, for example. The results of the model show that the number of tires that are detected as fake is 235. The number of frames detected as real is 83. The reason for identifying 83 of the frames as real is that these frames contain features (behavior) similar to the real features of Barack Obama. The percentage of this video being fake is 73.89937%. The percentage of frames that contain features similar to the features of the real behavior of Obama is 26.10062%, as shown in Table (4.41).

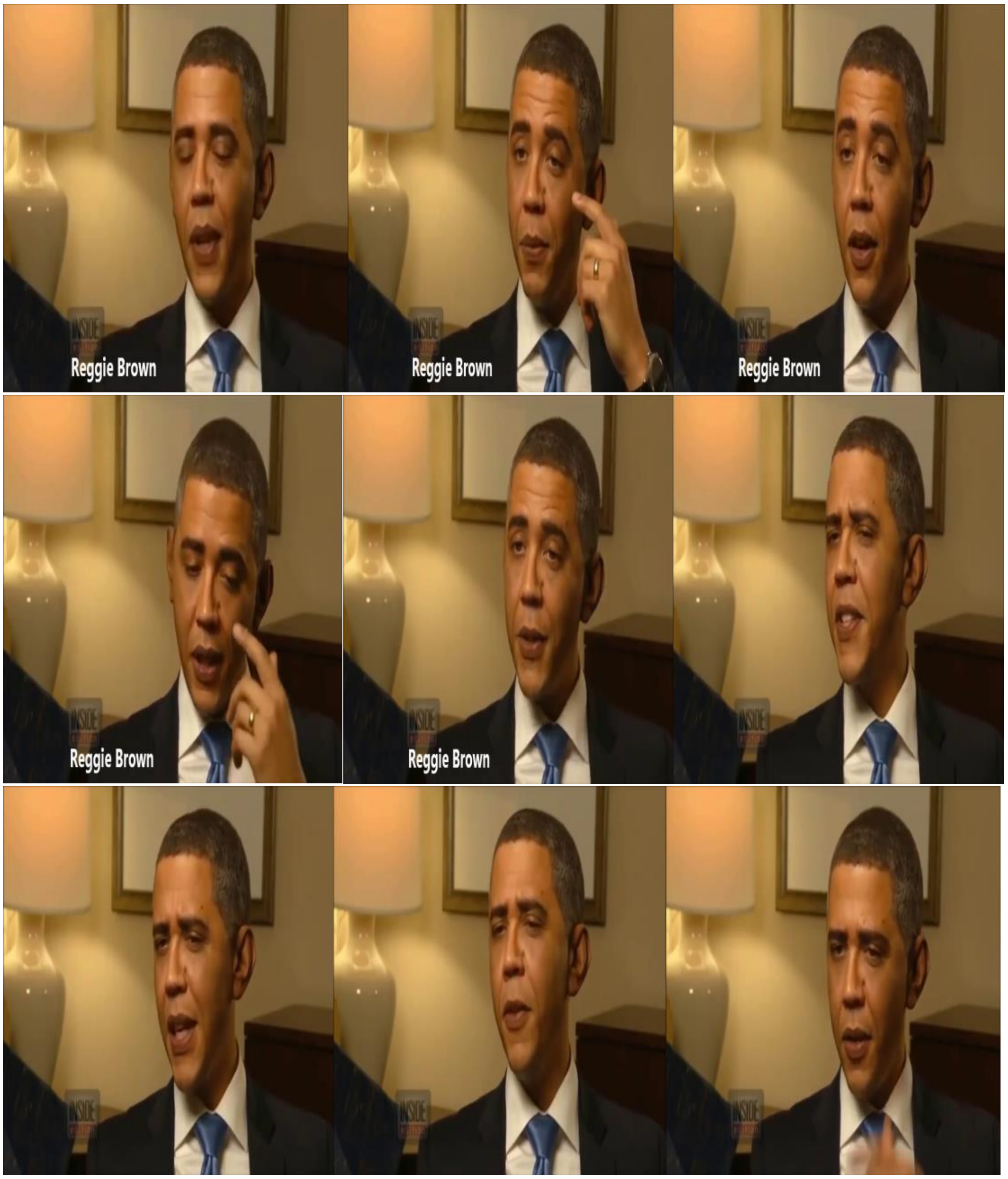
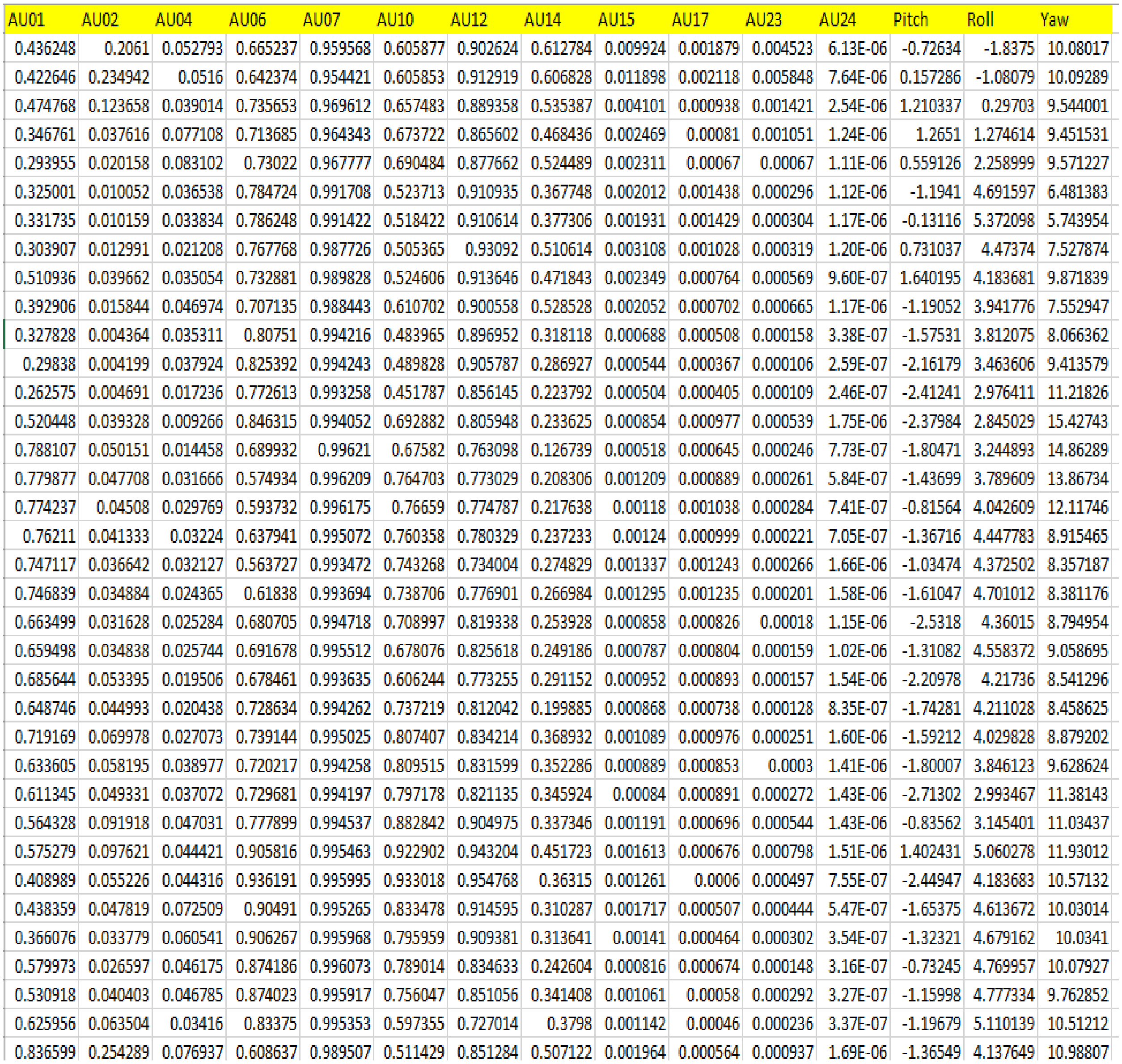


Figure (4.19): The Fake Video of Barack Obama

Table (4.40): The Extracted Features from The Fake Video of Barack Obama



AU01	AU02	AU04	AU06	AU07	AU10	AU12	AU14	AU15	AU17	AU23	AU24	Pitch	Roll	Yaw
0.436248	0.2061	0.052793	0.665237	0.959568	0.605877	0.902624	0.612784	0.009924	0.001879	0.004523	6.13E-06	-0.72634	-1.8375	10.08017
0.422646	0.234942	0.0516	0.642374	0.954421	0.605853	0.912919	0.606828	0.011898	0.002118	0.005848	7.64E-06	0.157286	-1.08079	10.09289
0.474768	0.123658	0.039014	0.735653	0.969612	0.657483	0.889358	0.535387	0.004101	0.000938	0.001421	2.54E-06	1.210337	0.29703	9.544001
0.346761	0.037616	0.077108	0.713685	0.964343	0.673722	0.865602	0.468436	0.002469	0.00081	0.001051	1.24E-06	1.2651	1.274614	9.451531
0.293955	0.020158	0.083102	0.73022	0.967777	0.690484	0.877662	0.524489	0.002311	0.00067	0.00067	1.11E-06	0.559126	2.258999	9.571227
0.325001	0.010052	0.036538	0.784724	0.991708	0.523713	0.910935	0.367748	0.002012	0.001438	0.000296	1.12E-06	-1.1941	4.691597	6.481383
0.331735	0.010159	0.033834	0.786248	0.991422	0.518422	0.910614	0.377306	0.001931	0.001429	0.000304	1.17E-06	-0.13116	5.372098	5.743954
0.303907	0.012991	0.021208	0.767768	0.987726	0.505365	0.93092	0.510614	0.003108	0.001028	0.000319	1.20E-06	0.731037	4.47374	7.527874
0.510936	0.039662	0.035054	0.732881	0.989828	0.524606	0.913646	0.471843	0.002349	0.000764	0.000569	9.60E-07	1.640195	4.183681	9.871839
0.392906	0.015844	0.046974	0.707135	0.988443	0.610702	0.900558	0.528528	0.002052	0.000702	0.000665	1.17E-06	-1.19052	3.941776	7.552947
0.327828	0.004364	0.035311	0.80751	0.994216	0.483965	0.896952	0.318118	0.000688	0.000508	0.000158	3.38E-07	-1.57531	3.812075	8.066362
0.29838	0.004199	0.037924	0.825392	0.994243	0.489828	0.905787	0.286927	0.000544	0.000367	0.000106	2.59E-07	-2.16179	3.463606	9.413579
0.262575	0.004691	0.017236	0.772613	0.993258	0.451787	0.856145	0.223792	0.000504	0.000405	0.000109	2.46E-07	-2.41241	2.976411	11.21826
0.520448	0.039328	0.009266	0.846315	0.994052	0.692882	0.805948	0.233625	0.000854	0.000977	0.000539	1.75E-06	-2.37984	2.845029	15.42743
0.788107	0.050151	0.014458	0.689932	0.99621	0.67582	0.763098	0.126739	0.000518	0.000645	0.000246	7.73E-07	-1.80471	3.244893	14.86289
0.779877	0.047708	0.031666	0.574934	0.996209	0.764703	0.773029	0.208306	0.001209	0.000889	0.000261	5.84E-07	-1.43699	3.789609	13.86734
0.774237	0.04508	0.029769	0.593732	0.996175	0.76659	0.774787	0.217638	0.00118	0.001038	0.000284	7.41E-07	-0.81564	4.042609	12.11746
0.76211	0.041333	0.03224	0.637941	0.995072	0.760358	0.780329	0.237233	0.00124	0.000999	0.000221	7.05E-07	-1.36716	4.447783	8.915465
0.747117	0.036642	0.032127	0.563727	0.993472	0.743268	0.734004	0.274829	0.001337	0.001243	0.000266	1.66E-06	-1.03474	4.372502	8.357187
0.746839	0.034884	0.024365	0.61838	0.993694	0.738706	0.776901	0.266984	0.001295	0.001235	0.000201	1.58E-06	-1.61047	4.701012	8.381176
0.663499	0.031628	0.025284	0.680705	0.994718	0.708997	0.819338	0.253928	0.000858	0.000826	0.00018	1.15E-06	-2.5318	4.36015	8.794954
0.659498	0.034838	0.025744	0.691678	0.995512	0.678076	0.825618	0.249186	0.000787	0.000804	0.000159	1.02E-06	-1.31082	4.558372	9.058695
0.685644	0.053395	0.019506	0.678461	0.993635	0.606244	0.773255	0.291152	0.000952	0.000893	0.000157	1.54E-06	-2.20978	4.21736	8.541296
0.648746	0.044993	0.020438	0.728634	0.994262	0.737219	0.812042	0.199885	0.000868	0.000738	0.000128	8.35E-07	-1.74281	4.211028	8.458625
0.719169	0.069978	0.027073	0.739144	0.995025	0.807407	0.834214	0.368932	0.001089	0.000976	0.000251	1.60E-06	-1.59212	4.029828	8.879202
0.633605	0.058195	0.038977	0.720217	0.994258	0.809515	0.831599	0.352286	0.000889	0.000853	0.0003	1.41E-06	-1.80007	3.846123	9.628624
0.611345	0.049331	0.037072	0.729681	0.994197	0.797178	0.821135	0.345924	0.00084	0.000891	0.000272	1.43E-06	-2.71302	2.993467	11.38143
0.564328	0.091918	0.047031	0.777899	0.994537	0.882842	0.904975	0.337346	0.001191	0.000696	0.000544	1.43E-06	-0.83562	3.145401	11.03437
0.575279	0.097621	0.044421	0.905816	0.995463	0.922902	0.943204	0.451723	0.001613	0.000676	0.000798	1.51E-06	1.402431	5.060278	11.93012
0.408989	0.055226	0.044316	0.936191	0.995995	0.933018	0.954768	0.36315	0.001261	0.0006	0.000497	7.55E-07	-2.44947	4.183683	10.57132
0.438359	0.047819	0.072509	0.90491	0.995265	0.833478	0.914595	0.310287	0.001717	0.000507	0.000444	5.47E-07	-1.65375	4.613672	10.03014
0.366076	0.033779	0.060541	0.906267	0.995968	0.795959	0.909381	0.313641	0.00141	0.000464	0.000302	3.54E-07	-1.32321	4.679162	10.0341
0.579973	0.026597	0.046175	0.874186	0.996073	0.789014	0.834633	0.242604	0.000816	0.000674	0.000148	3.16E-07	-0.73245	4.769957	10.07927
0.530918	0.040403	0.046785	0.874023	0.995917	0.756047	0.851056	0.341408	0.001061	0.00058	0.000292	3.27E-07	-1.15998	4.777334	9.762852
0.625956	0.063504	0.03416	0.83375	0.995353	0.597355	0.727014	0.3798	0.001142	0.00046	0.000236	3.37E-07	-1.19679	5.110139	10.51212
0.836599	0.254289	0.076937	0.608637	0.989507	0.511429	0.851284	0.507122	0.001964	0.000564	0.000937	1.69E-06	-1.36549	4.137649	10.98807

Table (4.41): Results of the Fake Video Example

Count of frames fake	235
Count of frames real	83
Percentage of this video is fake	73.89937%
Percentage of this video is real	26.10062%

4.12 Results Discussion

The method depends mainly on facial expressions and face pose, which represent human behavior. Each person has a special behavior of facial expression and face pose. This behavior differs from one person to another. Facial features represent facial expressions action units and facial poses. The more accurate they are, the more accurate this leads at distinguishing the real videos from the fake ones. Action unit and face pose are the main features in a person's behavior that distinguish him from others. The technique that extracts these features plays a major role in the accuracy of the model. The most accurate model for extracting features is (JAA-Net and hybrid face pose), which has been proven through experiments. When using the two models (JAA-Net and hybrid face pose) together in the method, they gave the highest results, and the reason is to link the facial expressions with the movement of the head, and here it created a new pattern for the person's behavior. Linking face pose and facial expressions proved that some action units are generated in a specific face pose, and this is a unique feature of behavior.

The results of this study are discussed as follows:

- One of the factors affecting the accuracy of the method is the selection of the K-fold cross-validation. As the best K-fold cross-validation gave good results when it is equal to five or six. When the value of K-fold cross-validation is less than five, it gives results with little accuracy. Also, when the K-fold cross-validation is greater than six, it maintains accuracy without increasing the accuracy of the method. In addition to that, the number of epochs is also affected. When the number of epochs is less than twenty, it has less accuracy. Also, when the value is greater than 100, the accuracy of the method remains stable and does not improve.

- Little accuracy is obtained when using the SVM model. SVM extracts action unit features from the face, and its value is either zero or one. This method in the extraction process is inaccurate, and the reason is that when the action value is greater than half, it is considered one, and when it is less than half, it is considered zero. The SVM model's work is sometimes considered inefficient in the action-unit extraction process.
- The features of face pose, when used in the prediction process, do not give high accuracy in detecting fake and real videos. Experiments have proven that when adding a facial pose to the rest of the models, it increases the accuracy of the method.
- The logistic model is better than the SVM model in terms of method accuracy. Also, the logistic model remains less accurate than the JAA-Net model.
- The best model prediction is BiLSTM in terms of accuracy. This model demonstrated the ability to predict fake and real videos compared to other models, where the accuracy is 99.403%. As for the LSTM model, it is less accurate than the BiLSTM model, as the accuracy is 92.19%. The rest of the predicted GRU and ANN models are between 87.15% and 93.46%. The high accuracy of BiLSTM is due to its bidirectional prediction ability, which increases the efficiency of the prediction process.

Chapter five

CONCLUSION AND FUTURE WORKS

CHAPTER FIVE
CONCLUSION AND FUTURE WORKS

5.1 Conclusions

The development of video creation techniques has made it difficult to identify fake and real videos. This dissertation is based on facial expressions. The features that are extracted from the facial expressions are the action unit and the face pose, and it is considered one of the most important features that distinguish the person's behavior. There are several conclusions that can be drawn as a result of this study:

1. Performing data preprocessing, especially removing frames that don't have features and using SMOTE to treat feature imbalance, leads to better prediction.
2. The model that extracts action unit features from facial expressions also affects the prediction. Some models extract the action unit while giving the value and quality of the facial expression. The best models for extracting features from facial expressions are JAA-Net and hybrid facial poses, through which distinctive features are obtained. This most distinctive feature helps the prediction model detect fake and real videos. Action unit occurrence values are confined between zero and one. Some models, such as SVM, specify the value of the action unit as either zero or one without giving it a fractional value (percentage), because the basis of their work is binary classification. SVM is the least accurate model in the proposed system.
3. Extracting features from videos is the most important stage of the proposed system. These features determine the person's behavior in real videos, which differs in fake videos. Action units are the features extracted

from facial expressions, and there are many types of action units. The selection of certain types of action units greatly affects the accuracy of the system. Some types of actions are repeated by most people in emotional states. In the proposed system, special types of action units are selected, which are considered more distinctive and whose percentage differs from one person to another.

4. Face pose features have a significant impact on increasing the possibility of predicting real and fake videos. The three facial features (yaw, pitch, and roll) are also behaviors that distinguish one person from another. When using these features with action features, it will generate more distinctive behavior among people. Some action units, for example, have a large percentage value when facial pitch, yaw, or roll. Through experiments, it has been proven that the features of the face pose combined with the action unit greatly increase the efficiency of the predictive model.
5. The BiLSTM prediction model that is used in the method proved to be highly efficient in predicting real and fake videos. The high efficiency of the BiLSTM model is due to the fact that it is bidirectional, which is the characteristic that enables it to predict past and future behavior. The other models had less efficiency, and this is due to the mechanism of their work in the prediction.
6. Using the three optimization methods of early stopping, dropout, and batch normalization, the following was found:
   - Deep learning uses the early stopping strategy to enhance model generalization. It entails keeping track of a model's performance as it is being trained. The goal is to identify the ideal training stage at which

the model has acquired sufficient knowledge to generalize correctly to new data without overfitting the training set.

- The dropout layer is a regularization method used in deep learning to prevent neural networks from overfitting. Each neuron in the dropout layer has a chance (dropout rate) of being "dropped out" or set to zero during training. This lessens the co-adaptation of neurons and prevents overfitting, making the network less sensitive to the presence or absence of any one neuron.
- Deep learning uses batch normalization as a strategy to enhance neural network generalization and training. It makes the optimization process more stable and speeds convergence by normalizing the input of each layer in a mini-batch during training.

7. The proposed system proved to have high accuracy compared to traditional methods. Traditional methods rely on artifacts to detect fake videos. One of the artifacts that the traditional methods are looking for is the inconsistency of the movement of the eyes, the wrapping of the new face on the old face, and other types of artifacts. But when tested on perfectly created videos, the traditional methods do not give high accuracy because those videos do not contain artifacts. While the proposed system does not depend on artifacts, it depends on the person's behavior by extracting features from facial expressions.

5.2 Future Works

There are some suggestions for future work:

1. The proposed system can be developed by using voice and facial expressions to predict real and fake videos.

2. The proposed system can be developed by using other types of models to extract the features of facial expressions.
3. It is possible to develop the system to work in real time when detecting fake and real videos.
4. The system is being trained on a dataset for world leaders that will help detect fake news and fake videos.
5. Improve the proposed prediction model using the microexpressions feature. Microexpressions are extremely brief and involuntary facial expressions that occur in a fraction of a second. They are often subtle and can reveal a person's true emotions, even when they are trying to conceal or control them.
6. The system can also be developed by working on adding features other than the action unit and the face pose that may increase the efficiency of the system, such as skeletons. The term skeleton often refers to a simplified model of the human body, represented by joints and limbs. This skeletal representation is used to track movement, posture, and gestures.
7. The system can also be developed on a person's body language, extract features from it, and train a prediction model to detect fake videos. Body language plays a significant role in interpersonal interactions as well as in understanding and interpreting the intentions of others.

REFERENCES

References

[1] Chen Qian2 Chen Change Loy Liming Jiang1 Ren Li2 Wayne Wu1, "DeeperForensics 1.0 A Large Scale Dataset for Real World Face Forgery Detection" pp. 2001– 2015,2020, doi:10.48550/arXiv.2001.03024.

[2] W. J. Hadi, S. M. Kadhem, and A. R. Abbas, "Fast discrimination of fake video manipulation," vol. 12, no. 3, pp. 2582–2587, 2022, doi: 10.11591/ijece.v12i3.pp2582-2587.

[3] Y. Mirsky and W. Lee, "The creation and detection of deepfakes: A survey," *ACM Computing Surveys (CSUR)*, vol. 54, no. 1, pp. 1–41, 2021.

[4] J. Teuwen *et al.*, "Deepfakes: temporal sequential analysis to detect face-swapped video clips using convolutional long short-term memory," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 13, no. 3, pp. 1–20, Mar. 2023, doi: 10.1007/s11263-020-01378-z.

[5] X. Yang, Y. Li, and S. Lyu, "Exposing Deep Fakes Using Inconsistent Head Poses," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2019-May. pp. 8261–8265, 2019. doi: 10.1109/ICASSP.2019.8683164.

[6] P. Gupta, K. Chugh, A. Dhall, and R. Subramanian, "The eyes know it: FakeET- An Eye-tracking Database to Understand Deepfake Perception," *ICMI 2020 - Proceedings of the 2020 International Conference on Multimodal Interaction*. pp. 519–527, 2020. doi: 10.1145/3382507.3418857.

[7] T. T. Nguyen *et al.*, "Deep Learning for Deepfakes Creation and Detection: A Survey," Sep. 2019, [Online]. Available: http://arxiv.org/abs/1909.11573

[8] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, “Springer,” *Information Fusion,* vol. 64. pp. 131–148, 2020. doi: 10.1016/j.inffus.2020.06.014.

[9] H. Qi *et al.*, “DeepRhythm: Exposing DeepFakes with Attentional Visual Heartbeat Rhythms,” Jun. 2020, [Online]. Available: http://arxiv.org/abs/2006.07634

[10] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, “Least Squares Generative Adversarial Networks,” in *Proceedings of the IEEE International Conference on Computer Vision,* Institute of Electrical and Electronics Engineers Inc., Dec. 2017, pp. 2813–2821. doi: 10.1109/ICCV.2017.304.

[11] J. Donahue, P. Krähenbühl, and T. Darrell, “Adversarial Feature Learning,” May 2016, [Online]. Available: http://arxiv.org/abs/1605.09782

[12] G. Zheng, R. Horstmeyer, C. Yang, G. Zheng, and C. Yang, “BiGAN,” *Iclr,* vol. 7, no. 9. p. 18, 2017. [Online]. Available: http://eutils.ncbi.nlm.nih.gov/entrez/eutils/elink.fcgi?dbfrom=pubmed&id=25243016&retmode=ref&cmd=prlinks%5Cnpapers3://publication/doi/10.1038/nphoton.2013.187%5Cnhttp://dx.doi.org/10.1038/nphoton.2013.187%5Cnhttp://www.nature.com/doifinder/10.1038/nphoto

[13] T. Shen, R. Liu, J. Bai, and Z. Li, “‘Deep Fakes’ using Generative Adversarial Networks (GAN),” *Noiselab UCSD*. 2018.

[14] O. Giudice, L. Guarnera, and S. Battiato, “Fighting deepfakes by detecting gan dct anomalies,” *Journal of Imaging,* vol. 7, no. 8. 2021. doi: 10.3390/jimaging7080128.

[15] E. Sanchez and M. Valstar, “Triple consistency loss for pairing distributions in

GAN-based face synthesis," Nov. 2018, [Online]. Available: http://arxiv.org/abs/1811.03492

[16] B. Dolhansky *et al.*, "The DeepFake Detection Challenge (DFDC) Dataset," Jun. 2020, [Online]. Available: http://arxiv.org/abs/2006.07397

[17] Bryan Lyon and Matt Tora," Exploring Deepfakes: Deploy powerful AI techniques for face replacement and more with this comprehensive guide", Packt Publishing, 2023

[18] R. Wang *et al.*, "FakeSpotter: A Simple yet Robust Baseline for Spotting AI-Synthesized Fake Faces," Sep. 2019, [Online]. Available: http://arxiv.org/abs/1909.06122

[19] R. Yasrab, W. Jiang, and A. Riaz, "Fighting Deepfakes Using Body Language Analysis," *Forecasting*, vol. 3, no. 2, pp. 303–321, 2021, doi: 10.3390/forecast3020020.

[20] M. Schuster, "Bidirectional recurrent neural networks," no. December 1997, 2016, doi: 10.1109/78.650093.

[21] D. Wodajo and S. Atnafu, "Deepfake Video Detection Using Convolutional Vision Transformer," Feb. 2021, [Online]. Available: http://arxiv.org/abs/2102.11126

[22] P. Korshunov and S. Marcel, "SUBJECTIVE AND OBJECTIVE EVALUATION OF DEEPFAKE VIDEOS." [Online]. Available: https://gitlab.idiap.ch/bob/bob.paper.subjective-deepfakes

[23] C. Ma, L. Chen, and J. Yong, "AU R-CNN: Encoding expert prior knowledge into R-CNN for action unit detection," *Neurocomputing*, vol. 355, pp. 35–47, 2019, doi: 10.1016/j.neucom.2019.03.082.

[24] L. Yao, Y. Wan, H. Ni, and B. Xu, "Action unit classification for facial expression recognition using active learning and SVM," *Multimedia Tools and Applications*, vol. 80, no. 16, pp. 24287–24301, 2021, doi: 10.1007/s11042-021-10836-w.

[25] U. A. Ciftci and I. Demir, "FakeCatcher: Detection of Synthetic Portrait Videos using Biological Signals," Jan. 2019, doi: 10.1109/TPAMI.2020.3009287.

[26] M. Allen, *Facial Action Coding System*. 2017. doi: 10.4135/9781483381411.n178.

[27] B. Zhu, H. Fang, Y. Sui, and L. Li, "Deepfakes for medical video de-identification: Privacy protection and diagnostic information preservation," *AIES 2020 - Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. pp. 414–420, 2020. doi: 10.1145/3375627.3375849.

[28] 2 Chen Qian2 Chen Change Loy Liming Jiang1 Ren Li2 Wayne Wu1, "DeeperForensics 1.0 A Large Scale Dataset for Real World Face Forgery Detection.pdf," vol. 7, no. 3. doi: 10.1007/s11063-008-9088-7.

[29] S. Ramachandran, A. V. Nadimpalli, and A. Rattani, "An Experimental Evaluation on Deepfake Detection using Deep Face Recognition," Oct. 2021, [Online]. Available: http://arxiv.org/abs/2110.01640

[30] S. Kaur, P. Kumar, and P. Kumaraguru, "Deepfakes: temporal sequential analysis to detect face-swapped video clips using convolutional long short-term memory," *Journal of Electronic Imaging*, vol. 29, no. 03, p. 1, 2020, doi: 10.1117/1.jei.29.3.033013.

[31] X. Yang, Y. Li, and S. Lyu, "Exposing Deep Fakes Using Inconsistent Head Poses," Nov. 2018, [Online]. Available: http://arxiv.org/abs/1811.00661

[32] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "Mesonet: a compact facial

video forgery detection network," in *2018 IEEE international workshop on information forensics and security (WIFS)*, IEEE, 2018, pp. 1–7.

[33] S. Agarwal, H. Farid, Y. Gu, M. He, K. Nagano, and H. Li, "Protecting World Leaders Against Deep Fakes," pp. 38-45 Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) , IEEE, 2019, Ed., Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) , IEEE, 2019, pp. 38-45.

[34] S. Agarwal, H. Farid, T. El-Gaaly, and S.-N. Lim, "Detecting deep-fake videos from appearance and behavior," in *2020 IEEE International Workshop on Information Forensics and Security (WIFS)*, IEEE, 2020, pp. 1–6.

[35] R. Tolosana, S. Romero-Tapiador, J. Fierrez, and R. Vera-Rodriguez, "DeepFakes Evolution: Analysis of Facial Regions and Fake Detection Performance," Apr. 2020, [Online]. Available: http://arxiv.org/abs/2004.07532

[36] T. Reolon Moreno D'incà, "DeepFake detection in videos SIV Project Report."

[37] A. Ismail, M. Elpeltagy, M. Zaki, and K. A. El-Dahshan, "Deepfake video detection: YOLO-Face convolution recurrent approach," *PeerJ Computer Science*, vol. 7, pp. 1–19, 2021, doi: 10.7717/PEERJ-CS.730.

[38] M. Boháček and H. Farid, "Protecting President Zelenskyy against Deep Fakes," pp. 1–8, 2022, [Online]. Available: http://arxiv.org/abs/2206.12043

[39] C. T. Jose, "Deepfake Detection using ImageNet models and Temporal Images of 468 Facial Landmarks," pp. 1–25, 2022, [Online]. Available: http://arxiv.org/abs/2208.06990

[40] B. Peng, C. Zhu, M. Zeng, and J. Gao, "Data augmentation for spoken language understanding via pretrained language models," *arXiv preprint arXiv:2004.13952*,

2020.

[41] A. Ismail, M. Elpeltagy, M. S. Zaki, and K. Eldahshan, "A new deep learning-based methodology for video deepfake detection using xgboost," *Sensors*, vol. 21, no. 16, Aug. 2021, doi: 10.3390/s21165413.

[42] W. Yang *et al.*, "AVoiD-DF: Audio-Visual Joint Learning for Detecting Deepfake," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 2015–2029, 2023.

[43] A. Hernández-García and P. König, "Further advantages of data augmentation on convolutional neural networks," in *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27*, Springer, 2018, pp. 95–103.

[44] X. Wang, Y. Sheng, H. Deng, and Z. Zhao, "CharCNN-SVM for Chinese text datasets sentiment classification with data augmentation," *International Journal of Innovative Computing, Information and Control*, vol. 15, no. 1, pp. 227–246, 2019.

[45] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-df: A large-scale challenging dataset for deepfake forensics," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 3207–3216.

[46] X. Wu, Z. Xie, Y. Gao, and Y. Xiao, "Sstnet: Detecting manipulated faces through spatial, steganalysis and temporal features," in *ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, IEEE, 2020, pp. 2952–2956.

[47] L. Chen, Y. Zheng, and J. Xiao, "Rethinking data augmentation for robust visual

question answering," in *European Conference on Computer Vision*, Springer, 2022, pp. 95–112.

[48] N. Takahashi, M. Gygli, B. Pfister, and L. Van Gool, "Deep convolutional neural networks and data augmentation for acoustic event detection," *arXiv preprint arXiv:1604.07160*, 2016.

[49] R. Mash, B. Borghetti, and J. Pecarina, "Improved aircraft recognition for aerial refueling through data augmentation in convolutional neural networks," in *Advances in Visual Computing: 12th International Symposium, ISVC 2016, Las Vegas, NV, USA, December 12-14, 2016, Proceedings, Part I 12*, Springer, 2016, pp. 113–122.

[50] G. G. Şahin and M. Steedman, "Data augmentation via dependency tree morphing for low-resource languages," *arXiv preprint arXiv:1903.09460*, 2019.

[51] J. Zhang, Y. Rao, C. Man, Z. Jiang, and S. Li, "Identification of cucumber leaf diseases using deep learning and small sample size for agricultural Internet of Things," *International Journal of Distributed Sensor Networks*, vol. 17, no. 4, p. 15501477211007408, 2021.

[52] Y. Huang, L. Lin, P. Cheng, J. Lyu, and X. Tang, "Lesion-based contrastive learning for diabetic retinopathy grading from fundus images," in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part II 24*, Springer, 2021, pp. 113–123.

[53] T. Araújo *et al.*, "Data augmentation for improving proliferative diabetic retinopathy detection in eye fundus images," *IEEE access*, vol. 8, pp. 182462–182474, 2020.

[54] J. Beinecke and D. Heider, “Gaussian noise up-sampling is better suited than SMOTE and ADASYN for clinical decision making,” *BioData Mining*, vol. 14, pp. 1–11, 2021.

[55] C. G. P. Srinivas, S. Balachander, Y. C. Singh Samant, B. V. Hariharan, and M. N. Devi, “Hardware Trojan detection using XGBoost algorithm for IoT with data augmentation using CTGAN and SMOTE,” in *Ubiquitous Communications and Network Computing: 4th EAI International Conference, UBICNET 2021, Virtual Event, March 2021, Proceedings*, Springer, 2021, pp. 116–127.

[56] D. W. Firmansyah and R. Sarno, “Data Augmentation Technique Using Two Step SMOTE for Electronic-nose Signal in Breath Ketone Level Detection.,” *International Journal of Intelligent Engineering & Systems*, vol. 16, no. 4, 2023.

[57] F. J. Moreno-Barea, J. M. Jerez, and L. Franco, “Gan-based data augmentation for prediction improvement using gene expression data in cancer,” in *International Conference on Computational Science*, Springer, 2022, pp. 28–42.

[58] T. Zhao *et al.*, “Graph data augmentation for graph machine learning: A survey,” *arXiv preprint arXiv:2202.08871*, 2022.

[59] M. Jampour and M. Javidi, “Multiview facial expression recognition, a survey,” *IEEE Transactions on Affective Computing*, vol. 13, no. 4, pp. 2086–2105, 2022.

[60] N. Zainudin *et al.*, “Horn Schunck Algorithm for Facial Expression Change Detection Classification,” *International Journal for Information Security Research*, vol. 5, no. 3, pp. 574–581, 2015, doi: 10.20533/ijisr.2042.4639.2015.0066.

[61] M. Karnati, A. Seal, D. Bhattacharjee, A. Yazidi, and O. Krejcar, “Understanding deep learning techniques for recognition of human emotions using facial

expressions: a comprehensive survey," *IEEE Transactions on Instrumentation and Measurement*, 2023.

[62] M. Sajjad *et al.*, "A comprehensive survey on deep facial expression recognition: challenges, applications, and future guidelines," *Alexandria Engineering Journal*, vol. 68, pp. 817–840, 2023.

[63] J. M. F. Dols and J. A. Russell, *The science of facial expression*. Oxford University Press, 2017.

[64] A. Hassouneh, A. M. Mutawa, and M. Murugappan, "Development of a real-time emotion recognition system using facial expressions and EEG based on machine learning and deep neural network methods," *Informatics in Medicine Unlocked*, vol. 20, p. 100372, 2020.

[65] M. J. A. I. Dujaili, "Survey on facial expressions recognition: databases, features and classification schemes," *Multimedia Tools and Applications*, pp. 1–22, 2023.

[66] M. Sajjad *et al.*, "A Comprehensive Survey on Deep Facial Expression Recognition : Challenges , Applications , and Future Guidelines," *Alexandria Engineering Journal*, vol. 68, pp. 817–840, 2023, doi: 10.1016/j.aej.2023.01.017.

[67] T. K. Ying-Li Tian and and Jeffrey F.Cohn, "Chapter 11. Facial Expression Analysis," *Journal of Infectious Diseases*, vol. 174, no. 4, pp. 835–838, 2013.

[68] P. Naga, S. Das Marri, and R. Borreo, "Facial emotion recognition methods, datasets and technologies: A literature survey," *Materials Today: Proceedings*, vol. 80, pp. 2824–2828, 2023.

[69] E. L. Rosenberg and P. Ekman, *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*. Oxford University Press, 2020.

[70] R. Zhi, M. Liu, and D. Zhang, “A comprehensive survey on automatic facial action unit analysis,” *The Visual Computer*, vol. 36, pp. 1067–1093, 2020.

[71] S. Jaiswal and M. Valstar, “Deep learning the dynamic appearance and shape of facial action units,” in *2016 IEEE winter conference on applications of computer vision (WACV)*, IEEE, 2016, pp. 1–8.

[72] Z. Shao, Z. Liu, J. Cai, Y. Wu, and L. Ma, “Facial Action Unit Detection Using Attention and Relation Learning,” *IEEE Transactions on Affective Computing*, vol. 13, no. 3, pp. 1274–1289, 2022, doi: 10.1109/TAFFC.2019.2948635.

[73] M. Kawulok, E. Celebi, and B. Smolka, *Advances in face detection and facial image analysis*. Springer, 2016.

[74] K. Khabarlak, “Fast Facial Landmark Detection and Applications: A Survey,” vol. 22, no. 1, pp. 12–41, 2022, doi: 10.24215/16666038.22.e02.

[75] D. Li, Z. Wang, Q. Gao, Y. Song, X. Yu, and C. Wang, “Facial expression recognition based on Electroencephalogram and facial landmark localization,” *Technology and Health Care*, vol. 27, no. 4, pp. 373–387, 2019.

[76] I. Journal, C. Vision, Y. Wu, and Q. Ji, “Fast Facial Landmark Detection and Applications: A Survey,” Journal of Computer Science and Technology, vol. 127, no. 22,2022.

[77] Y. Wu and Q. Ji, “Facial Landmark Detection: A Literature Survey,” *International Journal of Computer Vision*, vol. 127, no. 2, pp. 115–142, 2019, doi: 10.1007/s11263-018-1097-z.

[78] D. Canedo and A. J. R. Neves, “Facial Expression Recognition Using Computer Vision: A Systematic Review,” Applied Sciences, vol. 9, no. 21, p. 4678, Nov. 2019, doi: 10.3390/app9214678.

[79] K. Khabarlak and L. Koriashkina, “Fast Facial Landmark Detection and Applications: A Survey,” *Journal of Computer Science and Technology(Argentina)*, vol. 22, no. 1, pp. 12–41, 2022, doi: 10.24215/16666038.22.e02.

[80] V. Kazemi and J. S. Kth, “One Millisecond Face Alignment with an Ensemble of Regression Trees.” Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1867-1874, 2014

[81] D. Wanyonyi and T. Celik, “Open-Source Face Recognition Frameworks: A Review of the Landscape,” *IEEE Access*, vol. 10. Institute of Electrical and Electronics Engineers Inc., pp. 50601–50623, 2022. doi: 10.1109/ACCESS.2022.3170037.

[82] A. Sheka and V. Samun, “Boosting of Head Pose Estimation by Knowledge Distillation,” pp. 1–12, 2021, [Online]. Available: http://arxiv.org/abs/2108.09183

[83] V. Albiero, X. Chen, X. Yin, G. Pang, and T. Hassner, “img2pose: Face Alignment and Detection via 6DoF, Face Pose Estimation,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 7613–7623, 2021, doi: 10.1109/CVPR46437.2021.00753.

[84] T. Sha, W. Zhang, T. Shen, Z. Li, and T. Mei, “Deep Person Generation: A Survey from the Perspective of Face, Pose, and Cloth Synthesis,” *ACM Computing Surveys*, vol. 55, no. 12, pp. 1–37, 2023.

[85] M. Hassaballah and A. I. Awad, *Deep learning in computer vision: principles and applications*. CRC Press, 2020.

[86] A. Bhardwaj, W. Di, and J. Wei, *Deep Learning Essentials: Your hands-on guide to the fundamentals of deep learning and neural network modeling*. Packt

Publishing Ltd, 2018.

[87] N. Buduma, N. Buduma, and J. Papa, *Fundamentals of deep learning*. “ O’Reilly Media, Inc.,” O'Reilly Media, Inc,2022.

[88] L. Chen, *Deep learning and practice with mindspore*. Springer Nature, 2021.

[89] A. W. Trask, *Grokking deep learning*. Simon and Schuster, 2019.

[90] S. Minaee, A. Abdolrashidi, H. Su, M. Bennamoun, and D. Zhang, “Biometrics recognition using deep learning: a survey,” *Artificial Intelligence Review*, 2023, doi: 10.1007/s10462-022-10237-x.

[91] J. Ni, T. Young, V. Pandelea, F. Xue, and E. Cambria, “Recent advances in deep learning based dialogue systems: a systematic survey,” *Artificial Intelligence Review*, vol. 56, no. 4, pp. 3055–3155, 2023, doi: 10.1007/s10462-022-10248-8.

[92] J. Zipfel, F. Verworner, M. Fischer, U. Wieland, M. Kraus, and P. Zschech, “Anomaly detection for industrial quality assurance: A comparative evaluation of unsupervised deep learning models,” *Computers & Industrial Engineering*, vol. 177, p. 109045, 2023.

[93] K. G. Kim, “Book review: Deep learning,” *Healthcare informatics research*, vol. 22, no. 4, pp. 351–354, 2016.

[94] M. Kilickaya, J. van de Weijer, and Y. M. Asano, “Towards Label-Efficient Incremental Learning: A Survey,” *arXiv preprint arXiv:2302.00353*, 2023.

[95] S. Hardaha, D. R. Edla, and S. R. Parne, “A Survey on Convolutional Neural Networks for MRI Analysis,” *Wireless Personal Communications*, vol. 128, no. 2, pp. 1065–1085, 2023, doi: 10.1007/s11277-022-09989-0.

[96] L. Deng and Y. Liu, *Deep learning in natural language processing*. Springer,

2018.

[97] W. Hu *et al.*, "A state-of-the-art survey of artificial neural networks for Whole-slide Image analysis: From popular Convolutional Neural Networks to potential visual transformers," *Computers in Biology and Medicine*, vol. 161, 2023, doi: 10.1016/j.compbiomed.2023.107034.

[98] V. Zocca, G. Spacagna, D. Slater, and P. Roelants, *Python deep learning*. Packt Publishing Ltd, 2017.

[99] H. J. Jie and P. Wanda, "Runpool: A dynamic pooling layer for convolution neural network," *International Journal of Computational Intelligence Systems*, vol. 13, no. 1, pp. 66–76, 2020, doi: 10.2991/ijcis.d.200120.002.

[100] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[101] Dr. Juliansyah Noor, *Encyclopedia of Computational Biology Bioinformatics and*, vol. 53, no. 9. 2019.

[102] M. Sahu and R. Dash, *A survey on deep learning: Convolution neural network (cnn)*, vol. 153, no. January. Springer Singapore, 2021. doi: 10.1007/978-981-15-6202-0_32.

[103] K. Huang, A. Hussain, Q.-F. Wang, and R. Zhang, *Deep learning: fundamentals, theory and applications*, vol. 2. Springer, 2019.

[104] R. Ibrahim and M. O. Shafiq, "Explainable Convolutional Neural Networks: A Taxonomy, Review, and Future Directions," *ACM Computing Surveys*, vol. 55, no. 10, pp. 1–37, 2023.

[105] A. Patil and M. Rane, "Convolutional Neural Networks: An Overview and Its Applications in Pattern Recognition," *Smart Innovation, Systems and Technologies*, vol. 195, pp. 21–30, 2021, doi: 10.1007/978-981-15-7078-0_3.

[106] Teuwen, Jonas, and Nikita Moriakov. "Convolutional neural networks." *Handbook of medical image computing and computer assisted intervention.* Academic Press, 481-501, 2020.

[107] C. Kim *et al.*, “DESEM: Depthwise Separable Convolution-Based Multimodal Deep Learning for In-Game Action Anticipation,” *IEEE Access*, vol. 11, no. May, pp. 46504–46512, 2023, doi: 10.1109/ACCESS.2023.3271282.

[108] G. Huang, Y. Zhang, and J. Ou, “Transfer remaining useful life estimation of bearing using depth-wise separable convolution recurrent network,” *Measurement*, vol. 176, p. 109090, 2021.

[109] W. Ding, Z. Huang, Z. Huang, L. Tian, H. Wang, and S. Feng, “Designing efficient accelerator of depthwise separable convolutional neural network on FPGA,” *Journal of Systems Architecture*, vol. 97, pp. 278–286, 2019.

[110] V. V. Doan, D. H. Nguyen, Q. L. Tran, D. Van Nguyen, and T. H. Le, “Real-time image semantic segmentation networks with residual depth-wise separable blocks,” *Proceedings - 2018 Joint 10th International Conference on Soft Computing and Intelligent Systems and 19th International Symposium on Advanced Intelligent Systems, SCIS-ISIS 2018*, pp. 174–179, 2018, doi: 10.1109/SCIS-ISIS.2018.00037.

[111] Carion, Nicolas, et al. "End-to-end object detection with transformers." *European conference on computer vision. Cham: Springer International Publishing*, 2020.

[112] G. Zhao, Z. Zhang, H. Guan, P. Tang, and J. Wang, “Rethinking ReLU to Train Better CNNs,” *Proceedings - International Conference on Pattern Recognition*, vol. 2018-Augus, pp. 603–608, 2018, doi: 10.1109/ICPR.2018.8545612.

[113] B. Xu, N. Wang, T. Chen, and M. Li, “Empirical Evaluation of Rectified

Activations in Convolutional Network," May 2015, [Online]. Available: http://arxiv.org/abs/1505.00853

[114] H. Gholamalinezhad and H. Khosravi, "Pooling Methods in Deep Neural Networks, a Review," 2020, [Online]. Available: http://arxiv.org/abs/2009.07485

[115] VERSLOOT, Christian. "What are max pooling, average pooling, global max pooling and global average pooling?". *MachineCurve*, 2020.

[116] Q. Xu, M. Zhang, Z. Gu, and G. Pan, "Overfitting remedy by sparsifying regularization on fully-connected layers of CNNs," *Neurocomputing*, vol. 328, pp. 69–74, 2019, doi: 10.1016/j.neucom.2018.03.080.

[117] B. Mele and G. Altarelli, "Lepton spectra as a measure of b quark polarization at LEP," *Physics Letters B*, vol. 299, no. 3–4, pp. 345–350, 1993, doi: 10.1016/0370-2693(93)90272-J.

[118] F. Lei, X. Liu, Q. Dai, B. Wing, and K. Ling, "Shallow convolutional neural network for image classification," *SN Applied Sciences*, vol. 2, no. 1, pp. 1–8, 2020, doi: 10.1007/s42452-019-1903-4.

[119] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, "How does batch normalization help optimization?," *Advances in neural information processing systems*, vol. 31, 2018.

[120] A. Ucar, "Deep Convolutional Neural Networks for facial expression recognition," *Proceedings - 2017 IEEE International Conference on INnovations in Intelligent SysTems and Applications, INISTA 2017*, no. April, pp. 371–375, 2017, doi: 10.1109/INISTA.2017.8001188.

[121] M. Dixon, "Sequence classification of the limit order book using recurrent neural networks," *Journal of computational science*, vol. 24, pp. 277–286, 2018.

[122] M. Saraswat and Srishti, “Leveraging genre classification with RNN for Book recommendation,” *International Journal of Information Technology*, vol. 14, no. 7, pp. 3751–3756, 2022.

[123] Z. Shi, Y. Chen, and J. Cartlidge, “The lob recreation model: Predicting the limit order book from taq history using an ordinary differential equation recurrent neural network,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, pp. 548–556.

[124] M. Bhatia, S. Sharma, M. Hooda, and N. C. Debnath, “A Machine Learning Approach Toward Meal Classification and Assessment of Nutrients Value Based on Weather Conditions,” in *Big Data Analytics and Intelligence: A Perspective for Health Care*, Emerald Publishing Limited, 2020, pp. 223–241.

[125] T. Mikolov, A. Joulin, S. Chopra, M. Mathieu, and M. Ranzato, “Learning longer memory in recurrent neural networks,” *arXiv preprint arXiv:1412.7753*, 2014.

[126] D. Rengasamy, M. Jafari, B. Rothwell, X. Chen, and G. P. Figueredo, “Deep learning with dynamically weighted loss function for sensor-based prognostics and health management,” *Sensors (Switzerland)*, vol. 20, no. 3, 2020, doi: 10.3390/s20030723.

[127] T. Ahmad, J. Wu, H. S. Alwageed, F. Khan, J. Khan, and Y. Lee, “Human Activity Recognition Based on Deep-Temporal Learning Using Convolution Neural Networks Features and Bidirectional Gated Recurrent Unit With Features Selection,” *IEEE Access*, vol. 11, pp. 33148–33159, 2023.

[128] R. Dey and F. M. Salem, “Gate-variants of gated recurrent unit (GRU) neural networks,” in *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*, IEEE, 2017, pp. 1597–1600.

[129] M. Yang and J. Wang, “Adaptability of Financial Time Series Prediction Based on BiLSTM,” *Procedia Computer Science*, vol. 199, pp. 18–25, 2021, doi: 10.1016/j.procs.2022.01.003.

[130] A. Zahra, A. F. Hidayatullah, and S. Rani, “Bidirectional long-short term memory and conditional random field for tourism named entity recognition,” vol. 11, no. 4, pp. 1270–1277, 2022, doi: 10.11591/ijai.v11.i4.pp1270-1277.

[131] H. Gwon, C. Lee, R. Keum, and H. Choi, “Improvement in Network Intrusion Detection based on LSTM and Feature Embedding,” *Journal of KIISE*, vol. 48, no. 4, pp. 418–424, 2021, doi: 10.5626/jok.2021.48.4.418.

[132] L. Cai, S. Zhou, X. Yan, and R. Yuan, “A Stacked BiLSTM Neural Network Based on Coattention Mechanism for Question Answering,” vol. 2019, 2019.

[133] P. Anki and A. Bustamam, “Measuring the accuracy of LSTM and BiLSTM models in the application of artificial intelligence by applying chatbot programme,” vol. 23, no. 1, pp. 197–205, 2021, doi: 10.11591/ijeecs.v23.i1.pp197-205.

[134] S. Tam, R. Ben Said, and Ö. Ö. Tanriöver, “A ConvBiLSTM deep learning model-based approach for Twitter sentiment classification,” *IEEE Access*, vol. 9, pp. 41283–41293, 2021.

[135] Q. Yang, Y. Zhang, W. Dai, and S. J. Pan, *Transfer learning*. Cambridge University Press, 2020.

[136] D. Sarkar, R. Bali, and T. Ghosh, *Hands-On Transfer Learning with Python: Implement advanced deep learning and neural network models using TensorFlow and Keras*. Packt Publishing Ltd, 2018.

[137] Z. Shao, Z. Liu, J. Cai, and L. Ma, “J$\hat{\text{A}}$A-Net: Joint Facial Action

Unit Detection and Face Alignment via Adaptive Attention," Mar. 2020, doi: 10.1007/s11263-020-01378-z.

[138] Z. Shao, Z. Liu, J. Cai, and L. Ma, "Deep adaptive attention for joint facial action unit detection and face alignment," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 705–720.

[139] J. C. H. Ong, S. L. H. Lau, M.-Z. Ismadi, and X. Wang, "Feature pyramid network with self-guided attention refinement module for crack segmentation," *Structural Health Monitoring*, vol. 22, no. 1, pp. 672–688, 2023.

[140] Z. Liang, J. Shao, D. Zhang, and L. Gao, "Small object detection using deep feature pyramid networks," in *Advances in Multimedia Information Processing–PCM 2018: 19th Pacific-Rim Conference on Multimedia, Hefei, China, September 21-22, 2018, Proceedings, Part III 19*, Springer, 2018, pp. 554–564.

[141] X. Yang, W. Wang, J. Wu, C. Ding, and S. Ma, "MLA-Net : Feature Pyramid Network with Multi-Level Local Attention for Object Detection," pp. 1–13, 2022.

[142] REN, Shaoqing, et al. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*,vol. 28, pp. 1–14,2015.

[143] R. Elanwar, W. Qin, M. Betke, and D. Wijaya, "Extracting text from scanned Arabic books: a large-scale benchmark dataset and a fine-tuned Faster-R-CNN model," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 24, no. 4, pp. 349–362, 2021.

[144] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[145] H. Jiang and E. Learned-Miller, "Face detection with the faster R-CNN," in *2017*

12th IEEE international conference on automatic face & gesture recognition (FG 2017), IEEE, 2017, pp. 650–657.

[146] Q. Guo *et al.*, "SPARK: Spatial-aware Online Incremental Attack Against Visual Tracking," Oct. 2019, [Online]. Available: http://arxiv.org/abs/1910.08681

[147] S. Ayyachamy, V. Alex, M. Khened, and G. Krishnamurthi, "Medical image retrieval using Resnet-18," in *Medical imaging 2019: imaging informatics for healthcare, research, and applications*, SPIE, 2019, pp. 233–241.

[148] X. Ou *et al.*, "Moving object detection method via ResNet-18 with encoder–decoder structure in complex scenes," *IEEE Access*, vol. 7, pp. 108152–108160, 2019.

[149] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, "Multimodal machine learning: A survey and taxonomy," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 2, pp. 423–443, 2018.

[150] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *International journal of Remote sensing*, vol. 28, no. 5, pp. 823–870, 2007.

[151] A. Kurani, P. Doshi, A. Vakharia, and M. Shah, "A comprehensive comparative study of artificial neural network (ANN) and support vector machines (SVM) on stock forecasting," *Annals of Data Science*, vol. 10, no. 1, pp. 183–208, 2023.

[152] B. B. Hazarika, D. Gupta, and B. Kumar, "EEG signal classification using a novel universum-based twin parametric-margin support vector machine," *Cognitive Computation*, pp. 1–16, 2023.

[153] T. Schenkel, O. Ringhage, A. Márki, and J. Bae, "A COMPARATIVE STUDY OF FACIAL RECOGNITION TECHNIQUES With focus on low computational

power Bachelor Degree Project in Information Technology Basic level 30 ECTS Spring term 2019.”

[154] R. M. Aziz, M. F. Baluch, S. Patel, and P. Kumar, “A Machine Learning based Approach to Detect the Ethereum Fraud Transactions with Limited Attributes,” *Karbala International Journal of Modern Science*, vol. 8, no. 2, pp. 139–151, 2022, doi: 10.33640/2405-609x.3229.

[155] S. Fan and Z. Yang, “Towards objective human performance measurement for maritime safety: A new psychophysiological data-driven machine learning method,” *Reliability Engineering & System Safety*, vol. 233, p. 109103, 2023.

[156] K. Suzuki, *Artificial neural networks: Architectures and applications*. BoD–Books on Demand, 2013.

[157] H. Su, S. Liu, B. Zheng, X. Zhou, and K. Zheng, “A survey of trajectory distance measures and performance evaluation,” *The VLDB Journal*, vol. 29, pp. 3–32, 2020.

[158] V. Liermann and S. Li, “Methods of machine learning,” *The Digital Journey of Banking and Insurance, Volume III: Data Storage, Data Processing and Data Analysis*, pp. 225–238, 2021.

[159] D. Sarma, T. Mittra, and M. S. Hossain, “Personalized book recommendation system using machine learning algorithm,” *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 1, 2021.

[160] A. Marjuni, T. B. Adji, and R. Ferdiana, “Unsupervised software defect prediction using median absolute deviation threshold based spectral classifier on signed Laplacian matrix,” *Journal of Big Data*, vol. 6, pp. 1–20, 2019.

[161] R. Lippmann, “Book review:" neural networks, a comprehensive foundation", by

simon haykin," *International journal of neural systems*, vol. 5, no. 04, pp. 363–364, 2014.

[162] P. K. Kanti, P. Sharma, B. Koneru, P. Banerjee, and K. D. Jayan, "Thermophysical profile of graphene oxide and MXene hybrid nanofluids for sustainable energy applications: Model prediction with a Bayesian optimized neural network with K-cross fold validation," *FlatChem*, vol. 39, p. 100501, 2023.

[163] J.-H. Du, P. Patil, K. Roeder, and A. K. Kuchibhotla, "Extrapolated cross-validation for randomized ensembles," *arXiv preprint arXiv:2302.13511*, 2023.

[164] L. Jian, Z. Huang, J. Zhang, and Z. Hu, "Rapid Analysis of Cylindrical Bypass Flow Field Based on Deep Learning Model," in *IOP Conference Series: Earth and Environmental Science*, IOP Publishing, 2022, p. 12013.

مستخلص

التزييف العميق هو مصطلح يستخدم لوصف إنتاج الأفلام أو الأصوات أو الصور المصنعة أو المزيفة والتلاعب بها باستخدام أساليب الذكاء الاصطناعي (AI). هو يستلزم استخدام مجموعات بيانات ضخمة من الوسائط الحقيقية لتعلم نماذج الذكاء الاصطناعي، وخاصة شبكات الخصومة التوليدية (GANs).يمكن لهذه النماذج بعد ذلك إنتاج محتوى اصطناعي جديد يحاكي المحتوى الحقيقي ولكنه مزيف. هناك أنواع من التزييف العميق، بما في ذلك نقل تعابير الوجه، حيث يتم عرض تعبيرات وجه شخص ما على وجه شخص آخر في مقطع فيديو. فهو يتيح تغيير التعبيرات العاطفية للشخص، مما يجعله يظهر مشاعر غير تلك المسجلة. لانتحال شخصية شخص ما، يقوم لصوص الهوية بإنشاء أفلام تجعل هذا الشخص يبدو مختلفًا. تعتمد مشكلة الطرق التقليدية للكشف عن التزييف العميق على العناصر الموجودة في مقاطع الفيديو المزيفة. عندما تكون هناك مقاطع فيديو لا تحتوي على مؤثرات، يتم إنشاؤها بشكل قريب جدًا من مقاطع الفيديو الحقيقية، حيث أن الطرق التقليدية لا تعطي نتائج جيدة. الهدف من الأطروحة، الكشف عن التزييف العميق، يعتمد على سلوك الكائن من حيث تعبيرات الوجه في مقاطع الفيديو التي تم إنشاؤها بطريقة مثالية للتزييف. لكل شخص سلوكيات خاصة عند التحدث وتعابير الوجه مثل الحزن والغضب وغيرها. يمكن استغلال هذه الميزة للكشف عن التزييف العميق من خلال مقارنة سلوك الأشخاص باستخدام تعبيرات الوجه مثل وحدات حركة الوجه ووضعيات الوجه.

يتكون الطريقة المقترحة من مرحلتين رئيسيتين. المرحلة الأولى هي استخراج الملامح من تعابير الوجه ووضعيات الوجه. نأخذ الإطارات كمدخلات من مقاطع الفيديو الحقيقية والمزيفة ونستخرج ميزاتها. وأهم السمات المستخرجة من تعابير الوجه هي وضعيات الوجه ووحدات الحركة. وكل وحدة عمل عبارة عن حركة محددة لعضلات الوجه، وهي من السمات المهمة في تمييز سلوك الجسم. يقوم نموذج JAA-Net المدرّب مسبقًا باستخراج وحدة الحركة، إذ يبلغ عدد وحدات الحركة التي يستخرجها اثنتي عشرة وحدة. يستخرج نموذج Hybrid Face Pose المُدرب مسبقًا الميزات الثلاثة لوضعية الوجه، وهي الانعراج، والميل، والتدحرج. تتنبأ المرحلة الثانية بمقاطع الفيديو الحقيقية

والمزيفة باستخدام الميزات المجمعة من المرحلة الأولى ثم يتم إدخالها في نموذج التنبؤ. نموذج التنبؤ المستخدم هوBiLSTM ، والذي تم تدريبه باستخدام الميزات. بعد التدريب، يستطيع النموذج التنبؤ بما إذا كانت مقاطع الفيديو حقيقية أم مزيفة.

مجموعات البيانات المستخدمة في النظام المقترح هي مجموعة بيانات Barack Obama و Forensics++ . تحتوي مجموعة بيانات Barack Obama على مقاطع فيديو تعتبر فعالة وقريبة جدًا من الواقع. تم الاعتماد على مجموعة بيانات Barack Obama في عملية التدريب على نموذج التنبؤ. تم أيضًا استخدام مجموعة بيانات Forensics++ للتدريب ومقارنة النتائج التي تم الحصول عليها مع نتائج مجموعة بيانات Barack Obama. أثبت الطريقة المقترحة دقته العالية مقارنة بالطرق التقليدية، حيث بلغت دقة الطريقة المقترحة ٩٩,٤٠٣٪.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة بابل
كلية تكنولوجيا المعلومات
قسم البرمجيات

تنبؤ فيديو التزييف العميق القائم على تحليل سلوك الكائن باستخدام التعلم العميق

أطروحة مقدمة إلى

مجلس كلية تكنولوجيا المعلومات في جامعة بابل والتي هي جزء من متطلبات نيل درجة الدكتوراه فلسفة في تكنولوجيا المعلومات / البرمجيات

من قبل
قاسم جليل خضيرعبيس

بإشراف
أ.د. اسراء هادي علي

١٤٤٥هـ ٢٠٢٣م