



University of Kerbala
College of Computer Science & Information Technology
Computer Science Department

Verifying The Facial Kinship Evidence to Assist Forensic Investigation Based on Deep Neural Networks

A Thesis

Submitted to the Council of the College of Computer Science & Information Technology / University of Kerbala in Partial Fulfillment of the Requirements for the Master Degree in Computer Science

Written by

Ruaa Kadhim Khalaf Faraj

Supervised by

Assist. Prof. Dr. Noor Dhia Al-Shakarchy

2024 A.D.

1445 A.H.

بسم الله الرحمن الرحيم

(ولما بلغ أشده آتيناه حكما وعلما وكذلك نجزي المحسنين)

صدق الله العلي العظيم

سورة يوسف

الآية 22

Supervisor Certification

Supervisor Certification

I certify that the thesis entitled (**Verifying the Facial Kinship Evidence to Assist Forensic Investigation Based on Deep Neural Networks**) was prepared under my supervision at the department of Computer Science/College of Computer Science & Information Technology/ University of Kerbala as partial fulfillment of the requirements of the degree of Master in Computer Science.

Signature:

Supervisor Name: Asst. Prof. Dr. Noor D. Al-Shakarchy

Date: / /2024

The Head of the Department Certification

In view of the available recommendations, I forward the thesis entitled "**Verifying the Facial Kinship Evidence to Assist Forensic Investigation Based on Deep Neural Networks**" for debate by the examination committee.

Signature:


Assist. Prof. Dr. Muhannad Kamil Abdulhameed


Head of Computer Science Department


Date: / /2024


Certification of the Examination Committee

We hereby certify that we have studied the dissertation entitled (**Verifying the Facial Kinship Evidence to Assist Forensic Investigation based on Deep Neural Networks**) presented by the student (**Ruaa Kadhim Khalaf Faraj**) and examined him/her in its content and what is related to it, and that, in our opinion, it is adequate with (**Excellent**) standing as a thesis for the Master degree in Computer Science.

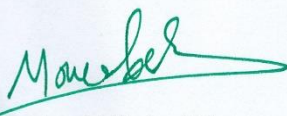
Signature: 
Name: Ahmed T. Sadiq
Title: Professor
Date: / / 2024
(Chairman)

Signature: 
Name: Ayad Hameed Mousa
Title: Assist. Prof. Dr.
Date: / / 2024
(Member)

Signature: 
Name: Hiba Jabbar Abdul wahid
Title: Assist. Prof. Dr.
Date: 6/3 / 2024
(Member)

Signature: 
Name: Noor Dhia Al-Shakarchy
Title: Assist. Prof. Dr.
Date: / / 2024
(Member and Supervisor)

Approved by the Dean of the College of Computer Science & Information Technology, University of Kerbala.

Signature: 
Assist. Prof. Dr. Mowafak khadom Mohsen
Date: / / 2024
(Dean of College of Computer Science & Information Technology)

Dedication

To

My family

My Father, My Mother,

My Dear Husband,

My Brothers, My Sisters,

My Lovely Sons.

Acknowledgement

First and foremost, praises and thanks to Allah, the Almighty, for his showers of blessings throughout my research work to complete the thesis successfully.

I would like to express my deep gratitude to my research supervisor, **Asst. Prof. Dr. Noor D. Al-Shakarchy**, for her efforts, scientific instructions and good advice, and her keenness to master every small and large, which had a great impact on enriching the thesis. And show it at this scientific level. Working and studying under her supervision was a great privilege and honor. I am very grateful for what you have given me, may Allah reward her.

I extend my sincere thanks and gratitude to Assist. Prof. Dr. Mowafak khadom Mohsen, the dean of the College, Assist. Prof. Dr. Muhsin Hasan, the Deputy Dean for Scientific Affairs, and Assist. Prof. Dr. Muhannad Kamil Abdulhameed, the Head of Computer Science Department, and all members of the teaching staff, and my colleagues are graduate students in the College of Computer Science and Information Technology / the University of Karbala.

Also, I wish to thank my friend for her assistance and for being there every time.

Ruaa Kadhim.

Abstract

The human face contains a wealth of information that is influenced by genetics, therefore family members often share common facial characteristics due to their kinship. Kinship verification is a challenging problem in many important applications, such as forensic science, biometrics, and face recognition. Kinship verification provides a powerful tool in forensic investigations, contributing to the resolution of missing person cases, social media analysis, genealogy research, and historical study. Although a DNA test is the most accurate mean for kinship verification, it unfortunately, cannot be used in many scenarios such as situations that require real-time processing or applications where we have non-cooperative users, and it is also costly. The main aim of this thesis to verify if two people have a kinship by analyzing two face images together, extracting the relationship features between them, and then determining if they have Kin or not.

The proposed system presents a kinship verification system based on automatically learning discriminative features from facial images using a three-dimensional convolutional neural network with a new architecture. This system consists of two main stages: the preprocessing stage and the kinship verification stage, and each stage includes multiple steps that perform different functions. In the preprocessing stage, the input images are prepared to be suitable for deep neural network model by scaling and normalizing them. The kinship verification stage is implemented to provide the kinship decision in two steps: the feature extraction step to extract the salient facial features as well as tracking these features in two input images, and then the classification step to decide on those images: kin or not.

The extensive experiments revealed that the 3D CNN had promising results compared with many state-of-the-art approaches. The accuracy of the proposed system reached 83.75% in the KinFaceW-I dataset, 91% in the KinFaceW-II dataset, and 75.5% in the FIW dataset.

Declaration Associated with this Thesis

Some of the works presented in this thesis have been published or accepted as listed below.

1. Ruaa Kadhim Khalaf, Noor D. AL-Shakarchy, "Verifying the Facial Kinship Evidence to Assist Forensic Investigation based on Deep Neural Networks", *International Conference on Emerging Trends and Applications in Artificial Intelligence (ICETAI – 2023)*, ISTANBUL MEDİPOL UNIVERSITY, TÜRKİYE.
2. Ruaa Kadhim Khalaf, Noor D. AL-Shakarchy," Facial Kinship Verification in Forensic Investigation Using Deep Neural Networks", *Journal of University of Kerbala, Iraq*,2024.
3. Ruaa Kadhim Khalaf, Noor D. AL-Shakarchy, "Verifying the Facial Kinship Evidence to Assist Forensic Investigation based on Deep Neural Networks: A review", *2023 International Conference on Information Technology Applied Mathematics Statistics, (ICITAMS-2023)*, Al-Qadisiyah University, Qadisiyah, Iraq.

Table of Contents

Dedication	i
Acknowledgement	ii
Abstract	iii
Declaration Associated with this Thesis	iv
Table of Contents	v
List of Tables	vi
List of Figures	vii
List of Abbreviations	ix
CHAPTER ONE: INTRODUCTION	
1.1 Introduction	1
1.2 Problem Statements	6
1.3 The Aim of the Thesis	6
1.4 Contributions	6
1.5 Thesis Organization	7
CHAPTER TWO: THEORETICAL BACKGROUND	
2.1 Overview	10
2.2 The Learning Process	11
2.3 Back Propagation (BP)	12
2.4 Deep Learning	14
2.4.1 Deep Convolutional Neural Networks (CNN)	16
2.4.2 Convolution Neural Networks Types	17
2.4.3 Three Dimensions deep Convolutional Neural Network (3D CNN)	19
2.4.4 Three Dimensions deep Convolution Neural Network (3D CNN): Architecture....	20
2.5 Region of Interest (ROI)	29
2.6 Scaling / Resizing Dimensionality.....	30
2.7 Normalization.....	31
2.8 Evaluation Metrics.....	32
2.9 Related Work	35
CHAPTER THREE: PROPOSED METHODOLOGY	
3.1 Overview	44

3.2 kinship Verification System Deployment stages	44
3.3 Proposed Model Architecture.....	48
3.3.1 Pre-processing stage.....	51
3.3.2 kinship verification stage.....	51
3.3.2.1 Feature Extraction Step.....	53
3.3.2.2 kinship verification Step.....	55
3.4 Model Evaluation.....	57
3.5 K-Folds Cross Validation.....	58
3.6 Models Implementation.....	
CHAPTER FOUR: RESULTS AND DISCUSSION	
4.1 Overview	61
4.2 System Requirement	61
4.3 Dataset.....	62
4.4 kinship Verification System Deployment stages.....	66
4.5 Comparison with State-of-the-Art Models.....	82
CHAPTER FIVE: CONCLUSION AND FUTURE WORKS	
5.1 Conclusion	86
5.2 Future Work	87
REFERENCES.....	89

List of Tables

Table 2.1: The Confusion Matrix	34
Table 2.2: The Related Works Summary	40
Table 3.1: The Proposed System.....	52
Table 4.1: The utilized 3D CNN Model's hyper-parameter values.....	72
Table 4.2: batch size tuning of 3D CNN model	73
Table 4.3: learning rate tuning of 3D CNN model	73
Table 4.4: The proposed model experimental results in training and testing set.....	74
Table 4.5: system's performance metrics for kinship verification on KinFace-II.....	75
Table 4.6: The Confusion Matrix CM.....	75
Table 4.7: The 10-fold cross-validation of proposed system in KinFaceW-II.....	77
Table 4.8: The evaluation measures values on KinFaceW-I	78
Table 4.9: The Confusion Matrix CM.....	78
Table 4.10: The 10-fold cross-validation of proposed system in KinFaceW-I	80
Table 4.11: The evaluation measures values on FIW dataset.....	81
Table 4.12: The Confusion Matrix CM	81
Table 4.13: A comparison with other state-of-the-art methods in KinFaceW-II	82
Table 4.14: A comparison with other state-of-the-art methods in KinFaceW-I	83
Table 4.15: A comparison with other state-of-the-art methods in FIW.....	83
Table 4.16: A The accuracy (%) on the eleven relationships of the FIW dataset	84

List of Figures

Figure 1.1: Common examples of kin relationships.....	2
Figure 1.2: general Facial Kinship Verification framework	5
Figure 2.1: An example of backpropagation in the neural network.....	14
Figure 2.2: The Performance of System with amount of data.....	15
Figure 2.3: 1D-CNN general architecture	18
Figure 2.4: Typical block diagram of 2D CNN	19
Figure 2.5: Typical block diagram of 3D CNN	20
Figure 2.6: Convolution Layer of 3D CNN.....	21
Figure 2.7: The multiplied and accumulated process of Convolution operation.....	22
Figure 2.8: ReLU plotted function	24
Figure 2.9: ReLU Functionality as a Pictorial Representation	24
Figure 2.10: Example of Max Pooling Operation	25
Figure 2.11: The Architecture of Fully Connected Layers.....	26
Figure 2.12: The dropout layer	27
Figure 2.13: Using 10 -Fold Cross-Validation Procedure.....	35
Figure 3.1: main stages of the proposed system.....	45
Figure 3.2: Architecture of the Proposed 3D CNN	47
Figure 3.3: Block Diagram of the proposed model	49
Figure 3.4: Batch Normalization sketch for simple network.....	54
Figure 3.5: The Max Pooling process.....	55
Figure 4.1: Part of the kinship pairs in KinFaceW-I.....	63
Figure 4.2: KinFaceW-II dataset	64
Figure 4.3: Example face pairs of each of the 11-relationship type in the FIW.....	66
Figure 4.4: The Data Set Division	67
Figure 4.5: Accuracy and loss functions of model using KinFaceW-II	69
Figure 4.6: Accuracy and loss functions of model using KinFaceW-I	70
Figure 4.7: Accuracy and loss functions of model using FIW.....	72

List of Abbreviations

Abbreviation	Description
3D CNN	Three- Dimensional Convolutional Neural Network
BN	Batch-Normalization
B-B	brother-brother
BP	Back Propagation
BSIF	Binarized Statistical Image Features
CM	Confusion Matrix
CNN	Convolution Neural Network
DNA	Deoxyribonucleic Acid
F-D	Father-Daughter
FKV	Facial Kinship Verification
FN	False Negative
FP	False Positive
F-S	Father-Son
GF-GD	grandfather-granddaughter
GF-GS	grandfather-grandson
GM-GD	Grandmother-granddaughter
GM-GS	Grandmother-grandson
HOG	Histograms of Oriented Gradient
K-NN	K-Nearest Neighbors
LBP	Local Binary Pattern
LPQ	Local Phase Quantization
M-D	Mother-Daughter

M-S	Mother-Son
ReLU	Rectified Linear Unit
ROI	Region Of Interest
SIBS	Brother-Sister
SIFT	Scale Invariant Feature Transform
S-S	Sister-Sister
SVM	Support Vector Machine
TN	True Negative
TP	True Positive

CHAPTER ONE
INTRODUCTION

1.1 Introduction

Forensic science is a multi-disciplinary that involves employing natural sciences such as computer, biology, chemistry, physics, and humanities such as psychology and sociology, in collecting and analyzing evidence left inside and outside the crime scene and employing the results of analyzes in determining the descriptions of the perpetrator, the victim, and the crime, to use these results later as evidence to convict or acquit the accused in court [1].

Face image analysis is a significant topic of study in image processing, computer vision, security, forensic investigation and pattern recognition at present. This is due to the fact that the human face includes huge social data including gender, age, and emotional state, as well as identifying features that may be utilized to determine an individual's identity. Face recognition, age estimation, facial expression recognition, and gender have been the subject of extensive study during the recent years [2],[3].

Computer-based kinship detection and verification is one of the study topics dependents on face image analysis. Kinship verification is a process of determining the biological relatedness between individuals. These relationships may be "Parent_Child", "Sibling_Sibling", "Grandparent_Child", etc., as seen in Figure 1.1. More than fifty percent of a parent's genes are passed on to their offspring. Children inherit several features from their parents, including resemblance in appearance, conduct, and voice, since genes overlap [2].

Facial Kinship Verification model uses face images or videos to automatically determine whether or not two people are related [4].



Figure 1.1: Common Examples of Kin Relationships[5].

While Deoxyribonucleic Acid (DNA) tests have revolutionized forensic science and becomes an invaluable tool in criminal investigations such as kinship verification, it is not without its limitations. Some of the main limitations of using DNA tests in forensic science include [6]:

- **Sample Contamination:** DNA testing requires handling small and sensitive samples. Contamination from external sources can occur during collection, handling, or analysis, leading to unreliable results.
- **Sample Degradation:** DNA samples can degrade over time, especially in adverse environmental conditions. This degradation can result in partial or low-quality DNA profiles, making it challenging to obtain conclusive results.

- **Mixtures and Low-Quantity Samples:** In crime scenes where multiple individuals' DNA is present, interpreting mixed DNA profiles can be complex and may not definitively identify the contributors.
- **Database Limitations:** DNA databases have limitations, such as incomplete or biased representation, which can affect the accuracy and reliability of matches and associations.
- **Ethical and Privacy Concerns:** The use of DNA in forensic investigations raises ethical questions about the collection, retention, and use of genetic information, as well as potential privacy issues.

These limitations make DNA tests unfeasible solutions in some life scenarios related to forensic applications and video surveillance. In addition, DNA needs many days to process, therefore it cannot be used in situations requiring real-time processing or with difficult users. Owing to the increasing expansion of multimedia, facial kinship verification has a substantial impact on a number of fields [5],[3].

Human sensory perception is insufficient to detect similarities between two photos taken by different people [7]. Poor accuracy rates are caused by the difficulty of quickly detecting face features including the size, shape, and color of face components. The most difficult and important step in the verification process is feature extraction, which is also the core of the system since the salient features made available for recognition have a direct impact on the precision of the kinship verification and recognition tasks [3],[8].

Fundamentally, there are two categories of limitations to recognizing kinship that have an impact on the accuracy of verifying facial kinship: **directly challenging** (associated to the kinship itself) which include differences in gender, age, and feature likeness amongst relatives, and **indirectly challenging** (relating to the database's environment) , such as lighting, noise, occlusion, facial expressions, blue, position variation, and lower picture quality, can have an impact on database photographs and lead to bad accuracy [2], [3], [9] .

In spite of FKV being an emerging, important, and challenging problem in computer vision, it has attracted a growing amount of interest, because of its potential and various applications in many fields such as:

1. In the field of public social security, it could be depended to find missing children, criminal investigations, and border control [10] .
2. In the social media domain, the FKV could be utilized in family photo album organization, improving the performance of face recognition systems and social media analysis [11] .
3. In the anthropology and genetics domain, FKV contributes in studying the hereditary characteristics of close relatives in social relationships [12].
4. FKV has potential applications in smart homes, the Internet of Things (IoT) [13]. furthermore, the FKV can be considered as an important research tool for visual kinship issues including family recognition and family retrieval [14].

The deep learning algorithm is one of the most popular ways of learning kinship. Deep Learning (DL) has now outperformed several shallow structural features (such as Histograms of Oriented Gradient (HOG), Scale Invariant Feature Transform (SIFT), and Local Binary Pattern (LBP)) and has achieved state-of-the-art results on important visual recognition functions. Deep learning generates more informative representations for classification tasks, resulting in higher accuracy [3].

The facial kinship verification method is made up of a number of stages, each of which has a number of steps that serve various purposes, as illustrated in Figure 1.2. The preprocessing stage (which covers all aspects of image preprocessing), classification stage (which covers feature extraction, feature selection, and other activities that might result in salient features, and classification task) [2] .



Figure 1.2: A General Facial Kinship Verification Framework[2]

1.2 Problem Statement

Facial Kinship Verification (FKV) means automatically determining whether two individuals have a kin relationship or not from their given face images or videos. Due to the fact that DNA requires several days for processing, it cannot be used in situations that require real-time processing or applications where we have non-cooperative users, and it is also costly.

1.3 The Aim of the Thesis

The thesis aims to develop and evaluate a facial kinship verification model that can accurately determine the kinship relationships between individuals based on their facial features. The main objectives of this thesis are:

1. Design new architecture for kinship verification system using Deep Neural Network.
2. The proposed system processes more than one input image in conjunction, and it is designed to learn related features between these face images. And this model can work in low-image-quality and unconstrained imaging environment.
3. To reach better accuracy for facial kinship verification.

1.4 Contributions

This thesis contributes to the development of a solid and usable Facial Kinship Verification system, which can be considered a crucial step with increasing interest in smart recognition and verification systems. The contribution of this thesis can be:

1. Building and implementing an automatic model that can process more than one image at the same time by using three-dimensional convolutional neural network (3D CNN). This model can detect the facial salient features as well as tracking these features through all input images and extract the related unique facial one in the same model and same time, which theoretically leads to reduce the processing time.
2. Coming up with a Kinship verification system based on face images, that is non-intrusive, cost-effective, and can be readily implemented in forensics investigation. And presenting a system, which offers an efficient implementation of low-quality and different illumination conditions or high-contrast images.

1.5 Thesis Organization

Chapter One: This chapter provides an introduction to the entire thesis, the contribution of thesis and the aim.

Chapter Two: Gives a full explanation of the main principles that will be used in the remainder of this thesis, as well as related work.

Chapter three: clarifies the main steps of designing the Facial Kinship verification model using Three-Dimensional Convolutional Neural Network.

Chapter four: The experimental results of verifying Facial Kinship are detailed in this Chapter.

Chapter five: The thesis's conclusion and future works will be discussed in this chapter.

CHAPTER TWO

THEORETICAL BACKGROUND

2.1 Overview

Kinship verification models are checking whether two persons belong to the same family or not. The goal of automatic kinship verification is to discover computational models that can accurately determine the existence of a familial relationship just by analyzing input patterns, including faces, voices, iris, and gaits. Kinship Verification has various practical applications in the real world, the most important in forensics investigations such as finding missing family members and family member identification, as well as, in social media analysis, genealogical, and historical research.

Although a DNA test is the most accurate method for determining kinship, it cannot be utilized in many situations, therefore Forensic Computing has become the reasonable solution in these situations. The existing works on verifying kinship basically share similar facial features. This involves employing shallow features LBP (Local Binary Patterns) and HOG (Histograms of Gradients) features as inputs to (Support Vector Machines) for verifying kinship such as (father, mother, brother, sister etc.).

These approaches perform better under some limited face image variations (resolution, illumination, blur etc.) but always suffer under uncontrolled environments or generalizing to unseen data. However, recent developments in machine learning indicate high performance, which can be reached from learned features based on deep learning methods rather than shallow features.

2.2 The Learning Process

The direction of signal propagation is one of the important topological choices when constructing a neural network, whether it is a forward or recurrent direction. Hence, there are two types of networks based on the direction of signal propagation; Feed-Forward Neural Network (FFNN), and Feed-Back Neural Network (FBNN) also called recurrent Neural Network. Combining the two topological features, (multi-layer ML and Feed-Forward FF), a general purposes neural network is constructed, and called Multi-Layer Feed-Forward network (MLFF). The learning is the process of the determination weights so that, the knowledge stored in the network as an experience function is modified, to allow a learning rule for changing the values of the weights. There are two types of neural networks according to the way learning is performed [15]:

- Fixed networks which the weights are fixed a priori according to the problem to solve and cannot be changed, i.e.: $\frac{dw}{dt} = 0$.
- Adaptive networks are can to change their weights, i.e.: $\frac{dw}{dt} \neq 0$

2.2.1 The Methods of adaptive ANN learning:

1. **Supervised learning:** some information is presented to the network during the learning process to determine the correct answer should be. Then the network is used the learning algorithm to adjust itself. The supervised learning paradigms include error-correction learning and stochastic learning. main objective of this learning is determining a set of weights that minimize the error between the desired and compute values [15]. Examples of these learning

algorithms are: Back Propagation (BP) and Support Vector Machine (SVM).

- 2. Unsupervised Learning:** In the unsupervised methods, no information can be presented to these networks and therefore cannot know exactly what the correct output should be. These networks must discover alone patterns, features, correlations, or Categories in the input data and code for them in the output. Examples of these learning algorithms found in the K-means clustering [16] .
- 3. Reinforcement learning:** In these methods, an indication of whether the output answer it computes is right or wrong is presented, so that a teacher is present, but the right answer is not presented to the network[15]. Examples of these learning is gaming and robotics.

In this thesis use Supervised learning.

2.3 Back Propagation Learning (BP)

It is a supervised learning-based method. It is a technique that enables the network to adjust its weights and biases to minimize the difference between predicted outputs and actual outputs. After choosing the weights of the network randomly, the back propagation algorithm is used to compute the necessary corrections [17], [18], [19]. The algorithm may be split into the following steps:

- 1. Feed-forward computation:** During the forward pass, input data is fed into the neural network. Each neuron's output is computed by apply an activation function to the weighted sum of its input.

2. **Calculate Error:** By comparing the predicted output of the neural network to the actual target output via a loss function, the difference between the two is calculated.
3. **Backpropagation:** The goal of backpropagation is to adjust the weights of the neurons in the network in such a way that the loss function is minimized. It works by propagating the error backward through the network to compute the gradients of the loss with respect to the weights.
4. **Gradient Calculation:** The gradient of the loss with respect to each weight and bias is calculated using the chain rule of calculus. This gradient indicates the direction and magnitude of change required to reduce the loss.
5. **Weight Update:** An optimization process, such as gradient descent, is used to update the weights and biases. It aims to move the network's parameters in the direction that reduces the loss.

Backpropagation is a key component of training deep neural networks. It allows networks to learn complex patterns in data by iteratively adjusting their parameters based on the errors observed during training. recent advances in optimization techniques, network architectures, and hardware have made it possible to train complex neural networks effectively.

When the value of the loss function becomes sufficiently minimal, the BP algorithm is ended [17]. Back Propagation algorithm illustrate in Figure 2.1.

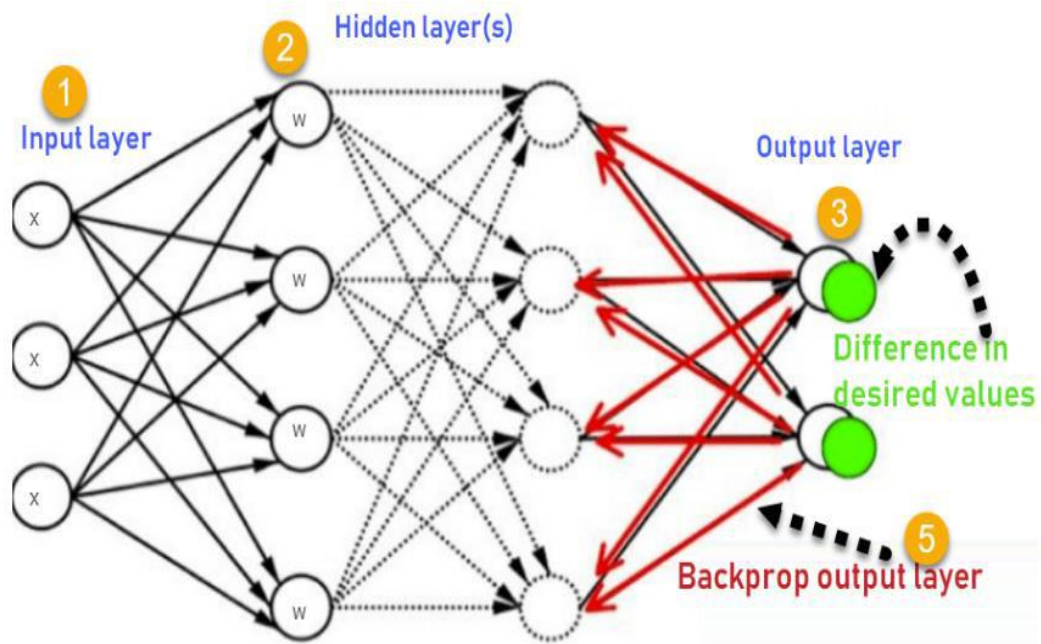


Figure 2.1: An Example of Backpropagation in The Neural Network

2.4 Deep learning

Deep learning is a branch of machine learning that deals with methods for modeling neurons in the human brain. It focuses on building and training artificial neural networks for feature learning and pattern classification [20] [21].

Deep learning models, often referred to as deep neural networks, consist of multiple layers of artificial neurons called nodes or units. Each layer receives inputs from the previous layer and applies a series of mathematical operations to generate outputs [22].

DL algorithms are relied essentially on the notion of ANN artificial neural networks. The availability of rich data and suitable processing power has made training such algorithms easy in today's society. As more data is collected, the performance of deep learning models continues to improve. A better representation of this can be seen in Figure 2.2, [23].

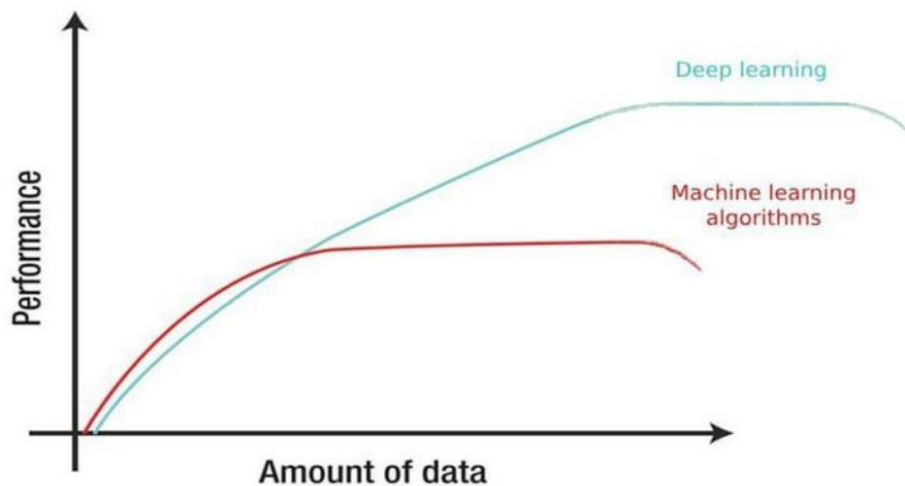


Figure 2.2: The Performance of System with Amount of Data

Deep learning architectures are used in several domains such as computer vision, image processing, natural language processing and many others. The deep learning model can automatically extract features, learns these extracted features automatically also whether they are visible to the eye or not as part of the training process, and produces correct results. But in the machine learning model, the features must design manually that capture relevant information from the data [24], [25].

Deep neural networks have multiple hidden layers of interconnected neurons, allowing them to automatically learn features from raw data. This enables the network to capture increasingly complex patterns and representations as information flows through the layers. Deep neural networks contain more layers than traditional shallow neural networks [26].

2.4.1 Deep Convolutional Neural Networks (CNN)

Convolution is one of the most important mathematical operations that can be applied widely in speech processing, image processing as well as in video processing to represent the modification of the shape based on another one which is called filters or kernel. This concept is exploited in deep neural networks and presented in Convolution Neural Networks CNN that can filter the inputs in deeper filters to extract efficient features [27].

CNN performs two major functions respectively, the feature extraction function and the classification function of these features, each of these functions utilizes some layers to achieve its specific purpose. The feature extraction function can be implemented by stacking convolution, nonlinear (activation), and pooling layers in some manner. The classification function is done through some fully connected (dense) and nonlinear (activation) layers. In addition, some generalization layers were added in two functions [27]. For each function, these layers are stacked as follows:

- **Feature Extraction Function**

CNN's feature extractor architecture consists of one or more convolution layers, each followed by a nonlinear layer representing the activation function applied to that layer to distinguish the special signal of useful features on each hidden layer. The convolution layer is responsible for extracting and constructing the feature maps of the input image. In

other words, the convolution layer acts as a local filter on the input image, with the filter kernel coefficients set during the training process. [28].

A non-linear function is employed in both general neural networks and CNN; for recognizing the special signal of important features on each hidden layer. CNN can implement a number of non-linear layers (activation functions), including 'Rectified Linear Units (ReLUs) and Tanh [27],[29] .

Using a subsampling (pooling) layer, the CNNs model is more robust against noise and blurring conditions. It is responsible for reducing the resolution of the features of an input image. This reduction in feature resolution is done by combining the outputs of a neuron, which are clusters at one layer, into a single neuron in the next layer using one of the pooling functions such as MAX pooling, MIN pooling or AVERAGE pooling[29].

- **Classification function**

Finally, a standard neural network with one or more hidden layers can be concatenated with dense or fully connected (FC) layers. The last layer of the model (also the most latest FC layer) represents the output layer where classification decisions are made[27].

2.4.2 Convolution Neural Networks Types:

There are three types of Convolution Neural Networks (CNN) in the real world based on the input dimensions used. these are one-dimensional, two-dimensional, and 3-dimensional CNNs [30].

A 1D Convolutional Neural Network (1D CNN) is a type of neural network architecture primarily designed to process one-dimensional sequential data, like time series data, audio signals, and text data.it can extract spectral features from the data [30],[31]. The general 1D-CNN architecture can be seen in Figure 2.3 as follows.

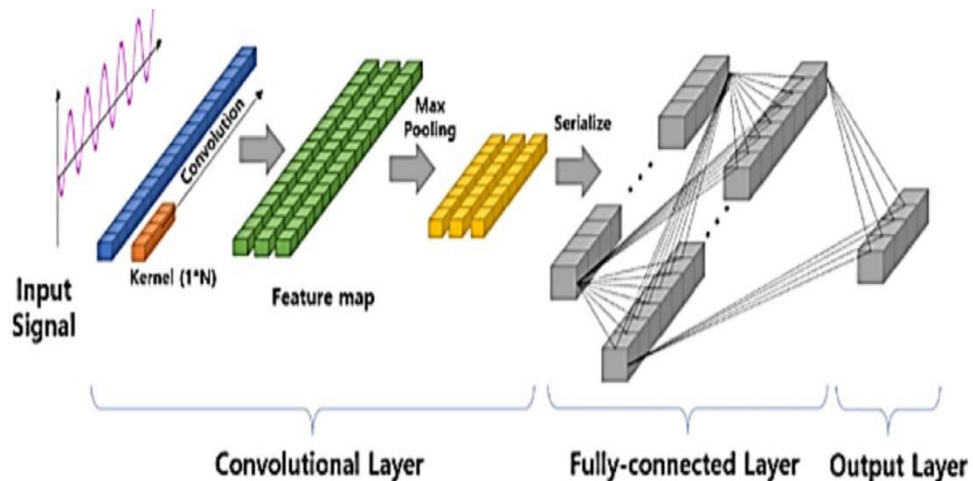


Figure 2.3.: 1D-CNN General Architecture

2D CNNs are the most common type of CNNs, it can extract spatial features from the input data. 2D CNNs are designed to process two-dimensional data, which is typically represented as images. Images, have two dimensions: width and height. 2D CNNs perform convolution operations across both dimensions and often have multiple channels, allowing them to capture different features in the data [26]. Figure 2.4 is illustrating the typical block diagram of 2D CNNs.

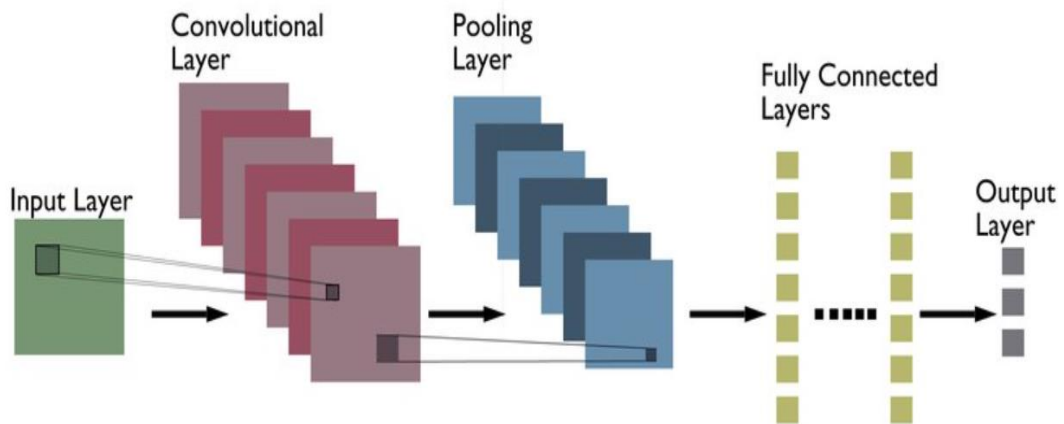


Figure 2.4.: Typical Block Diagram of 2D CNN

2.4.3 Three Dimensions Deep Convolutional Neural Network (3D CNN)

A 3D-CNN is based on the concept of convolutional neural networks (CNNs) but with the adding of a temporal dimension. 3D-CNN are specifically designed for processing three-dimensional data, such as videos and medical scans (e.g., MRI, CT) [30].

Three Dimension deep Convolutional Neural network (3D CNN) is implemented to consider the spatio-temporal information in the processing of video applications such as Video classification [27]. All 2-dimensions Classification methods deal with video by ignoring the temporal features of that video. The classification is based on the frames of video without taking into account the changes in dynamic features over time and the predictions for the next frames. In other words, the methods based on a frame cannot utilize all available information in the sequences

of images. In these classifications, one frame is independently classified at a time. by using 2D CNN [32] [33].

The architecture of 3D CNN and its layers is similar to 2D architecture but uses 3D kernels instead of 2D. The 3D CNN consist of many layers performs a specific function and uses saved weights to produce its output. These layers consist of 3D convolutional layers, non linear layers, pooling layers, and fully connected layers [30], in addition, some layers can be added to provide a generalization to the network, these layers represented batch normalization layers and dropout layers. The block diagram of 3D CNN is shown in Figure 2.5.

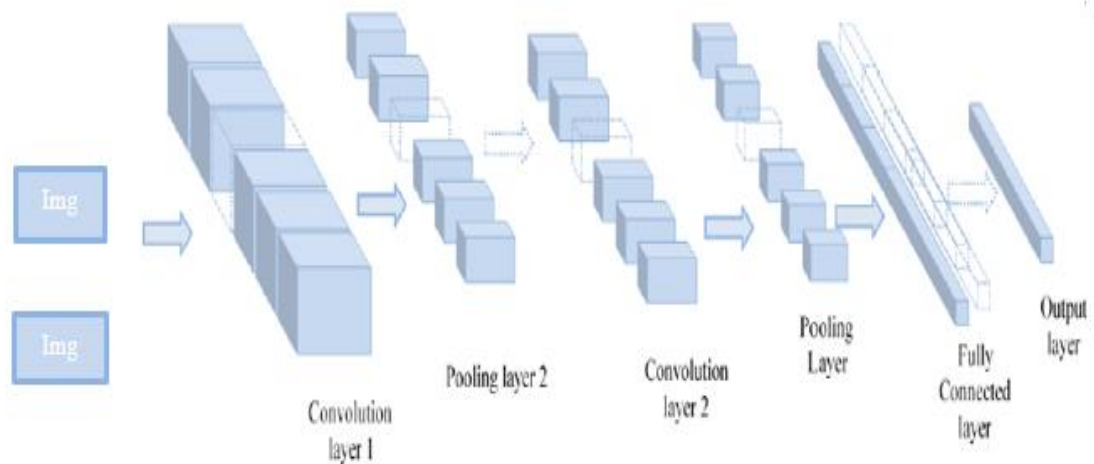


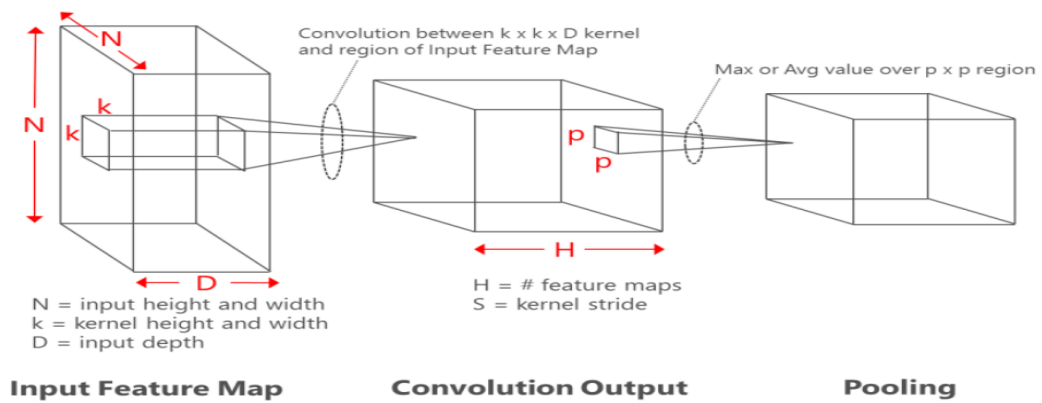
Figure 2.5: A Common Block Diagram of 3D CNN

2.4.4 Three Dimensions Deep Convolutional Neural Network (3D CNN): Architecture

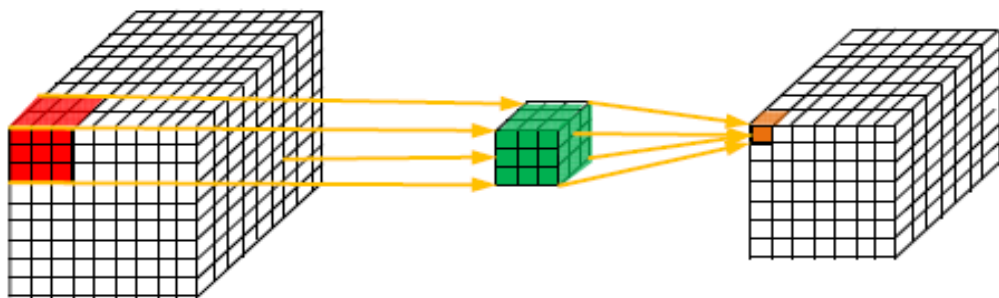
The layers of 3D CNN can be shown as:

a) Convolution Layer

The 3D convolution layer extracts features and produces feature maps by applying a 3D kernel (filter) to a sequence of images. The weights saved represented the kernel coefficient, which determines the training process. The 3D convolution layer is similar to 2D convolution layer in functions worked and in the hierarchy of feature maps extracted excepted 3D convolution layer used kernel stride of 3-dimensions [33] [34]. Figure 2.6 represents the process of 3D convolution utilized by 3D CNN.



(a)



(b)

Figure 2.6: Convolution Layer of 3D CNN (a)Pictorial Representation of convolution Process[35] (b) The 3D Convolution Process in The Convolution Layer [34]

A 3D input volume of dimensions $N \times N \times D$ is convolved with H kernels of dimension $k \times k \times D$ and stride S as shown in Figure 2.6(a).

H kernels process with a sliding window and starting from the top-left corner of the input to the bottom-right corner of the sequence of images, each kernel is moved from left to right in a downward direction, according to the value of stride, usually one element at a time so that convolved the input with one kernel generated the one output feature, so convolved input with H kernel generates H features independently [35].

Each feature in the output will contain elements which is multiplied and accumulated element-by-element to create one output feature. Figure 2.7 represents the multiplied and accumulated process with H kernel size $k \times k$.

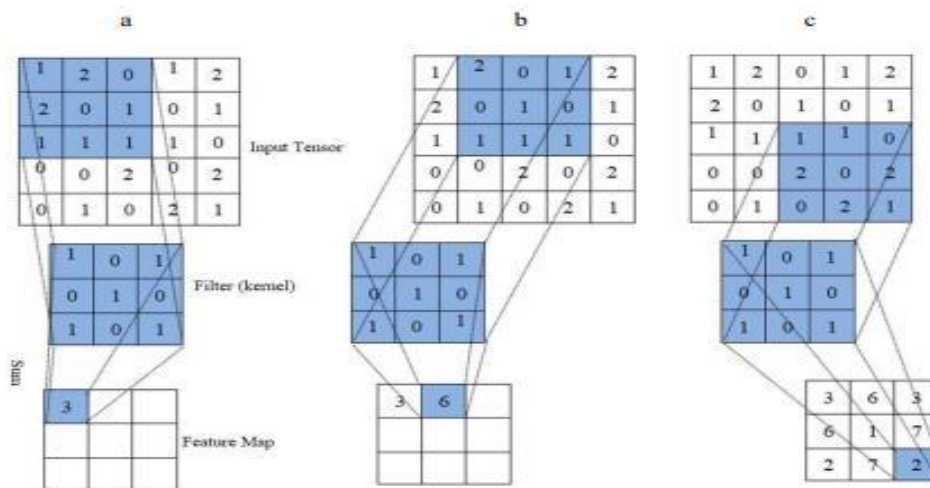


Figure 2.7: The Multiplied and Accumulated Process of Convolution Operation

Many control parameters which are called hyper parameters are determined in convolution layer to control the output size of the convolution layer [36]. These parameters can be detected as:

1. Zero-padding
2. Filter size
3. Number of filters.
4. Stride.

b) Non-Linear Layer

The activation function is a node in the network, that describes the method by which the input is converted into an output [26]. This work, uses two approaches, ReLU and sigmoid function.

Non-Linear layers apply many functions, including the 'Rectified Linear Units (ReLU) function' and 'sigmoid function'. These functions are used to determine the distinct features of each hidden layer. The feature maps output from the convolution layers becomes the input to Non-Linear layers to convert the linear output into nonlinear [29].

Rectified Linear Units (ReLU) is a non-linear "trigger" function used in both general neural networks and CNNs as an activation function consisting of the non-linear layer concatenated with the convolution layer, Figure 2.8 illustrates a plot of a ReLU function [37].

The ReLU function is applied to each input element as a threshold operation. As a result, all values less than zero are set to zero [38]. Thus, the ReLU is represented as follows:

$$relu(x) = \max(0, x) \dots \dots \dots (1)$$

The network trains many times faster with the ReLU when compared with other non-linear functions, therefore the ReLU function can be considered the best choice [39]. The ReLU functionality and corresponding transfer function plotted is illustrated in Figure 2.9.

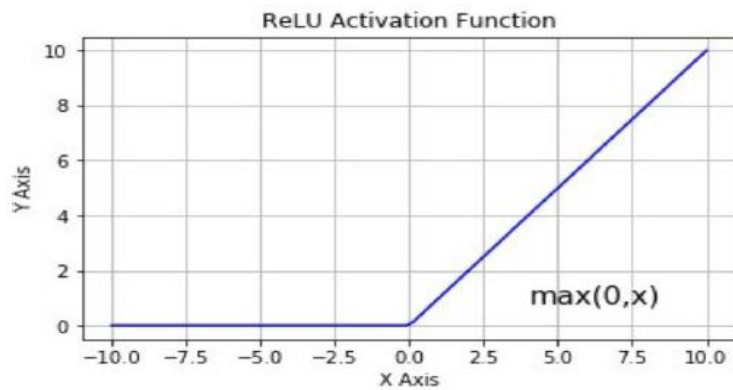


Figure 2.8: ReLU Plotted Function

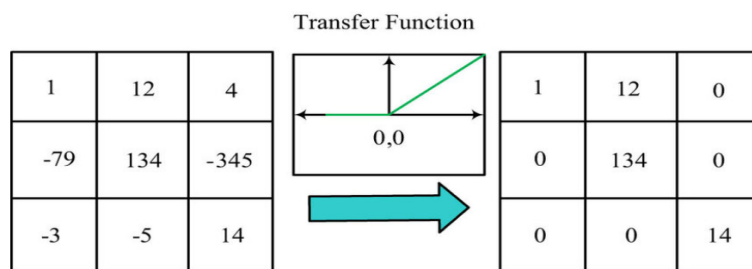


Figure 2.9: ReLU Functionality as a Pictorial Representation

c) Pooling Layer

The subsampling layer, which is another name for the pooling layer, reduces the resolution of the features to provide robustness against blur and noise. The reduction can be achieved on both a spatial and temporal dimension by combining several neuron clusters at one layer into a single neuron in the next layer. First, the input is divided into non-overlapping three-dimensional spaces, and then one of the pooling functions is applied [35],[40]. These functions done pooling are max pooling, mini pooling, and average pooling. Max\Mini pooling function selects the maximum\minimum value of the four values (clustering neurons) and average pooling functions calculate the average of the four values then averaging fraction rounds to the nearest integer in each region.

The 3D CNN model's pooling layer deals with 3D clusters of neurons (3D and non-overlapping spaces). [30] [41]. Figure 2.10 illustrate pooling process.

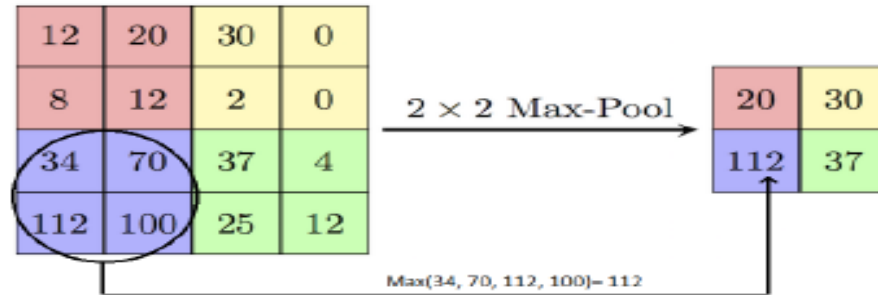


Figure 2.10: Example of Max Pooling Operation.

d) Fully Connected Layers

The last layers of CNN, called as fully connected layers, performed the classification phase. They have followed the convolution and pooling layers. These layers represent the traditional neural network which flattens the output of previous layers, which can be described mathematically by summing a weighting of the previous layer of features and then applying the specific activation function. The named fully connected comes from using all the elements of all the features of the previous layer in the computation of each element of each output feature. Figure 2.11 shows the general block diagram of the fully connected layer [18] [42].

The output layer is the final fully layer responsible for make decisions and producing the class score directly. The sigmoid function is a more suitable activation function for the output layer with binary classification, sigmoid function is mathematically expressed as:

$$F(x) = \frac{1}{e^{-x}} \dots \dots \dots (2)$$

when with multi-class classification purpose, the Softmax function is a more suitable activation function [39].

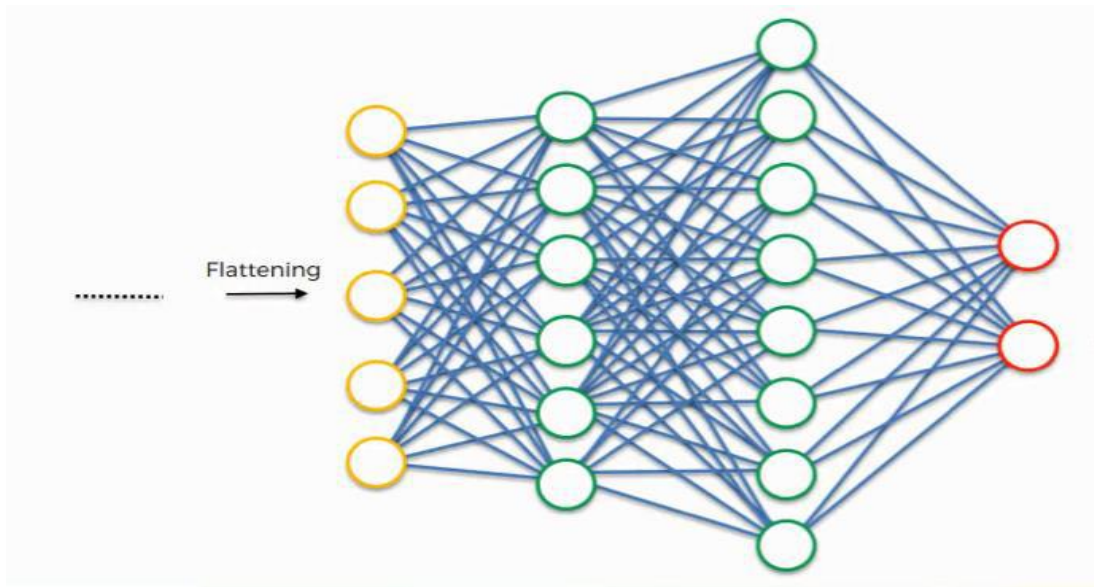


Figure 2.11: The Architecture of Fully Connected Layers

e) Dropout Layer

A dropout layer is a regularization technique commonly in deep learning models, to prevent overfitting[18]. It helps improve the generalization of the model by reducing the interdependence of neurons within a particular layer. To effectively control noise during the training process. The dropout refers to the dropping of some units into a neural network as illustrated in Figure 2.12. Dropping a unit randomly mean it is temporarily remove from the network with all its incoming and outgoing links [43].

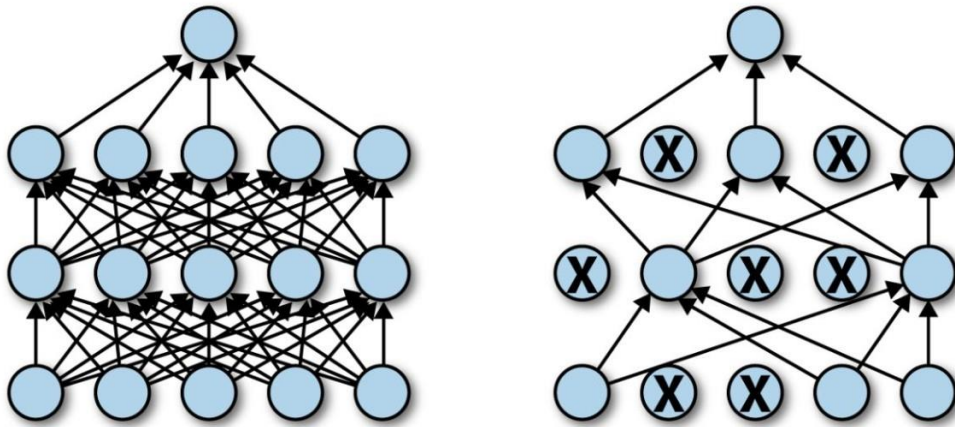


Figure 2.12: The Dropout Layer

f) Batch-Normalization (BN) Layer

The distribution of each input layer changes as the parameters of the previous layers change. The complexity of training deep neural networks is consequently increased. This slows down the training process by requiring lower learning rates and accurate parameterization, and therefore make training models very difficult. Internal covariate shift is the term used to describe this phenomenon, and fix it by normalizing layer inputs when building the normalization into the model architecture then give it strength, especially for each training mini-batch. Batch-Normalization enables the use of a lot higher learning rates with less care in initialization, and it's also eliminates the need to drop out in certain case, outperforming of the original model with a large margin [44] [45] [46] .

$$\mu_B = \frac{1}{m} \sum_{i=1}^m x_i \dots \dots \dots (3)$$

$$\sigma_B^2 = \frac{1}{m} \sum_{i=1}^m (x_i - \mu_B)^2 \dots \dots \dots (4)$$

Where, B: the mini batch. M: the size of mini batch. x_i : input. μ : mean σ^2 : variance.

For a d-dimensional input of a layer of the network, each dimension of its input is separately normalized,

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \dots \dots \dots (5)$$

ϵ is a very small value to avoid divided by zero.

Then to restore the representation power of the network apply the following transformation:

$$y_i = \gamma \hat{x}_i + \beta \dots \dots \dots (6)$$

Where the parameters γ and β are subsequently learnt in the optimization process.

The final activation from the unit with batch normalization can be get as[44]:

$$a^l = g^l (BN(W^l a^{[l-1]})) \dots \dots \dots (7)$$

Where, $BN(W^l a^{[l-1]})$: is y_i

W: is weights

[l]: is the layer in the network

$a^{[l-1]}$ is the input to a layer l

g^l : is non-linearity activation function of *layer*^[l]

Algorithm (2.1): Batch-Normalize BN Layers

Input: layer's input.

Output: normalized layer's output.

Step 1: compute the activation function of the layer's input by apply:

$$z^{[l]} = g(W^{[l]}a^{[l-1]})$$

Step 2: normalize z by do four equations as the following:

- a) Calculate mean (μ) of the mini-batch as equation (3).
- b) Calculate variance (σ^2) of the mini-batch as equation (4).
- c) Calculate \hat{x}_i by subtracting mean from z and subsequently dividing by standard deviation (σ). A small number, epsilon (ϵ), is added to the denominator to prevent divide by zero as equation (5).
- d) Calculate y_i by multiplying \hat{x}_i with scale γ and adding a shift β and use y_i place of a z of non-linearity (ReLU) input as equation (6).

Step 3: compute the final activation function by applying equation (7).

End.

2.5 Regions of Interest (ROI) Detection

(ROI) detection is a computer vision technique that involves extraction of regions of an image or video that are of particular interest or relevance for analysis or processing. Usually, these regions contain objects or information that are the focus of a particular task, based on the features (facial features) that the proposed system uses, the face is the most interesting part to use the proposed system on. (ROI) detection plays

important role in a variety of fields, such as object detection, medical imaging, and surveillance [47]. There are several methods of (ROI) such as

- Scale-Invariant Feature Transform (SIFT).
- Single Shot MultiBox Detector (SSD).
- Haarcascade
- Multi-task Cascaded Convolutional Networks (MTCNN).

Multi-task Cascaded Convolutional Networks (MTCNN) can detect faces with uncontrolled conditions images such as non-uniform illumination, height pose direction, face rotation, etc. [48].

2.6 Scaling\Resizing Dimensionality

Image resizing, also known as scaling, is one of the main operations in image processing to reduce or enlarge the image when doing resolution change [49].

Changing the number of pixels contained in the image leads to controls on the physical size of this image by reducing or enlarging it. When an image is resized, its pixel information is modified, so that a reduction in image size results in the deletion of any unneeded pixel information, and an increase in the size of the image causes it to generate new pixels and add new information based on its best estimation [49].

Image resizing employs Bi-Cubic Interpolation algorithm, and Bi-Cubic Interpolation equation:

$$f(x, y) = \sum_{i=0}^3 \sum_{j=0}^3 a_{ij} x^i y^j \dots\dots\dots (8)$$

a_{ij} : are the coefficients determined by solving a system of equations based on the values of the surrounding grid points and their derivatives.

i, j : from 0 to 3 in both dimensions, representing a 4x4 grid of neighboring points.

The equation can be expressed using the sixteen closest neighbors of point (x, y) to account for the sixteen coefficients from the sixteen equations in the sixteen unknowns, as in Equation (9).

$$f(x, y) = [x_3 \quad x_2 \quad x \quad 1] \begin{bmatrix} a_{3,3} & a_{3,2} & a_{3,1} & a_{3,0} \\ a_{2,3} & a_{2,2} & a_{2,1} & a_{2,0} \\ a_{1,3} & a_{1,2} & a_{1,1} & a_{1,0} \\ a_{0,3} & a_{0,2} & a_{0,1} & a_{0,0} \end{bmatrix} \begin{bmatrix} y_3 \\ y_2 \\ y \\ 1 \end{bmatrix} \dots\dots\dots (9)$$

2.7 Normalization

Normalization is a common data preparation technique used in deep learning that may be regarded as important, normalization is a process for changing the range of pixel intensity values with normally viewed. The main goal of normalization is to ensure that all values share a certain property. The values of pixels between 0 and 255 Presenting these raw format values to the neural network models, which deal with small weight values during input processing, may result in challenges during modeling such as when the model trains slower than expected. Therefore, prior preparation of the image pixel values is more beneficial[50].

There are many normalization techniques are there namely Min-Max normalization, Z-score normalization and Decimal scaling

normalization [51]. In this work, normalization is done by Z-score normalization as seen in equation:

$$Z = \frac{(X - \mu)}{\sigma} \dots\dots\dots (10)$$

Z: the new data.

X: original value.

μ : mean of data.

σ : standard deviation of data.

2.8 Evaluation Metrics

Evaluating any model is an essential part of any system. When evaluated with one metric, a model may produce a good skill score and satisfactory results, but when evaluated with other metrics, the results are unsatisfactory. Classification accuracy is used most of the times to measure the performance of a model, but this measure cannot represent the true judgment of a model. Therefore, multiple evaluation metrics, including Accuracy, Confusion Matrix, Logarithmic Loss, recall, F1 Score, Precision, and Mean Squared Error, must be utilized [52] [27].

2.8.1 Accuracy

Accuracy is determined by dividing the number of correctly identified samples by the total number of samples in the testing dataset [53]. According to equation (11).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \dots\dots\dots (11)$$

TP: True Positive, TN: True Negative, FP: False Positive, FN: False Negative.

2.8.2 Precision

Precision measures the proportion of correctly predicted positive instances out of the total instances predicted as positive by the model[54]. It was computed in equation (12).

$$\text{Precision} = \frac{\text{TruePositives}}{\text{TruePositives} + \text{FalsePositives}} \dots\dots\dots (12)$$

2.8.3 Recall

It is computed by dividing the number of true positives by the sum of true positives and false negatives. [53]. According to equation (13).

$$\text{Recall} = \frac{\text{TruePositives}}{\text{TruePositives} + \text{FalseNegatives}} \dots\dots\dots (13)$$

2.8.4 F1 Score

The F1 Score is calculated using precision and recall measures to determine the accuracy of a test. F1 Score ranges between 0 and 1, with the greater value indicating superior performance [54]. According to equation (14).

$$\text{F_score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \dots\dots\dots (14)$$

2.8.5 Confusion Matrix CM

Confusion Matrix is one of the most important and common methods to describe the complete performance of the model [52]. Table 2.1 represent the CM of binary classification model which have samples belongs to the two classes “YES or NO”, such that: True Positive (TP) is when the model predicted YES and the actual output was also YES, True Negative (TN) is when the model predicted NO and the actual output was NO, False Positive (FP) is when the model predicted NO and the actual output was YES, and False Negative (FN) is when the model predicted YES and the actual output was NO. [2].

Table 2.1: The Confusion Matrix

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

2.8.6 K-Cross Validation

Cross validation is a commonly employed method for evaluating the performance of a predictive model. It assists with estimating how well a model will perform on unseen data [55].

In K-fold cross-validation, the dataset is split into K subsets of equal size, or "folds". The model is trained and evaluated K times, each time, a different fold is used as the validation set while the remaining folds are used as the training set. Performance measurement, such as accuracy

or mean squared error, are computed each iteration. The final performance estimate is then obtained by averaging the results from all K iterations.

Overall, K-fold cross-validation is a valuable technique for model evaluation, helping in building more reliable and generalizable machine learning models. Figure 2.13 illustrates the process of 10-fold cross-validation.

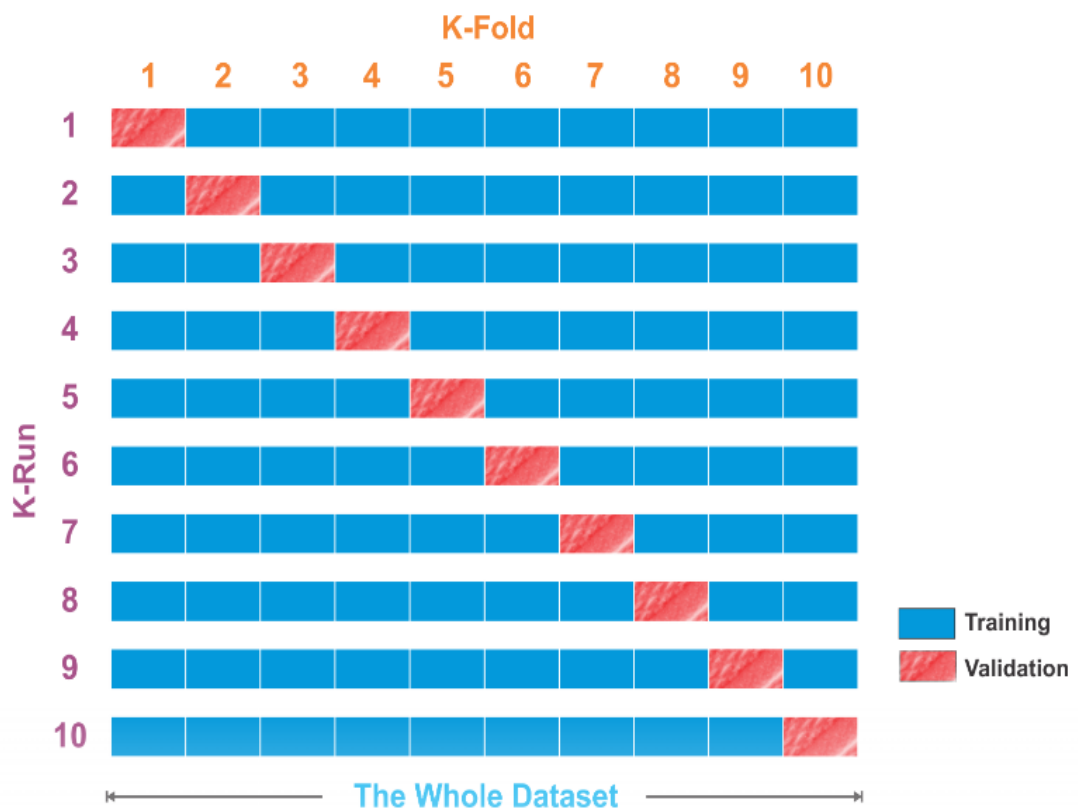


Figure 2.13: Using 10 -Fold Cross-Validation Procedure

2.9 Related Work

Facial kinship verification is an active area of research, with many studies focusing on the most effective techniques for extracting distinguishing characteristics used in existing kinship verification methods.

1. Xiaoting Wu et al.[56] proposed a Similarity Metric-based Convolution Neural Network (SMCNN) technique on KinFaceW-I and KinFaceW-II datasets for kinship verification. The SMCNN structure utilizes two identical Convolution Neural Networks, each with eight layers. The L1 norm between two CNN outputs was calculated, and a decision was made using a learned threshold. The superior results are obtained with KinFaceW-II due to the cropped image sharing a comparable environment, such as chrominance and brightness. The verification accuracies were 72.7% in the KinFaceW-I dataset and 79.25% in the KinFaceW-II dataset.
2. Chergui et al. [57] 2019 use (ResNet) for the feature extraction stage, in addition to our suggested pair feature extraction function and Rank Features (T test) to decrease the number of features through feature selection, and then uses the Support Vector Machine (SVM) to verify kinship decision. The verification accuracy was 87.16% on the Cornell Kin Face, 83.68% on the UBKin, 82.07% on the Family101, 79.76% on the KinFaceW-I, and 76.89% on KinFaceW-II datasets respectively.
3. Chergui et al. [58] in 2019 provided a method for extracting features that are based on combining several descriptors (Local Binary Patterns (LBP), Local Phase Quantization (LPQ), and Binarized Statistical Image Feature (BSIF)). The Multi-Block (MB) representation approach was used, and the effect of alternative feature representations for verifying kinship was examined to minimize the number of features selected using the TTest function. For kinship classification, a Support Vector

Machine (SVM) was used. This technique achieves kinship verification accuracy of 84.74% On Cornell KinFace, 82.74% on UBKin, 81.69% on KinFace-I, 80.12% on KinFace-II, and 78.16% on Family 101, respectively.

4. Nandy and Mondal [59] in 2019 proposed a Deep learning technique using Siamese Convolutional Neural Network Architecture for Facial Kinship Verification. And combine the two networks into a single output using a similarity metric and fully connected networks. Several similarity metrics were employed, including L1 norm, L2 norm, and Cosine Similarity, but the cosine similarity metrics outperforms than L1 and L2 metrics concerning accuracy because of effective and simple learning of the objective function. This network gives good accuracy. The verification accuracy was 67.66% on the FIW datasets.
5. Van and Hoang [60] in 2019 uses Local Binary Pattern(LBP) for Kinship on different color space. Then, they calculated features based on (Chi-Square) distance and applied the Fisher score to discover important features. A Support Vector Machine is utilized for model training and prediction. The verification accuracy for the KinFaceW-I and the KinFaceW-II databases was 72.6% and 81.8%, respectively.
6. Zhang et al.[61] in 2019 Deep learning techniques have been presented for their promising performance. Using shape and appearance complementary information. Both are necessary when determining kinship from face photos. To train this model with limited Kinship data, the researchers used an adaptation-based two-

- phase training approach using large-scale face recognition data, with the verification accuracy (78.3%) on the KinFaceW-I dataset.
7. Goyal & Meenpal [62] in 2019 used two descriptors (Local Binary Pattern(LBP) and Histogram of Gradient (HOG)) to identify salient features. Then, used a Support Vector Machine classifier to obtain an understanding of the retrieved face characteristics. The results showed that the (LBP_SVM) technique performed better than the (HOG_SVM) technique. On the KinFaceW-I dataset dataset, the LBP_SVM approach's mean accuracy was 75.57 %. On the KinFaceW-I dataset, the HOG_SVM technique had an average accuracy of 73.35 %.
 8. Mukherjee & Meenpal [63] in (2019) provided a method to increase the accuracy of verifying kinship that relies on a compound Local Binary Pattern(CLBP) and local feature-based discriminate analysis (LFDA). Long feature vectors were generated using these two techniques. The only methodology that accelerated the process and selected the best features was the entire feature vector-based LFDA feature selection method. A KNN classifier was used. They used the KinFaceW-I and KinFaceW-II datasets; the verification accuracy was 82.82 and 89.36, respectively.
 9. Felipe Crispim et al. [64] provided a technique of kinship verification using RGB-D Face Data in 2020. First, a database including 3D information and kinship annotations was collected, and depth information from normalized 3D reconstructions was combined with 2D images to create RGBD data. then, employ a Convolutional Neural Network and a Support Vector Machine for

classification and comparison. The CNN was evaluated on a commonly used two databases for kinship verification (KinFace W-I and KinFace W-II) and the Kin3D dataset. The results suggest that adding depth information enhances the model's performance, increasing classification accuracy to 90%.

10. Rachmadi et al. [65] uses a family-aware convolutional neural network (FA-CNN) to solve the problem of visual kinship verification in 2020. A family-aware network and a kinship verification network are added to the state-of-the-art facial recognition model to create the suggested classifier. The family-aware network adjusts its weights by learning family-specific features using deep metric learning loss, whereas the kinship verification network uses softmax loss to learn the kinship verification issue. The average accuracy is 68.84% on the FIW dataset.

11. Wu et al. [66] in 2021 presented a technique for localizing multiple facial feature points by employing a facial feature detector to extract SIFT descriptors around each facial feature point in a face picture. In conclusion, two methods, feature combination and distance metric learning, are employed to verify the relationship between two face photos, the verification accuracy was 73.8% on the KinFace W-I and 78.23% on the KinFace W-II datasets respectively.

12. Yan and Song [67] proposed developed a Deep relational network that uses multi-scale information from face photos to verify kinship in 2021. Their model utilized two Convolutional Neural Networks

with shared parameters for every pair of face images in order to extract distinct scales of facial features. Their network used the benefits of convolutional computations to turn different parts of a face into different scale features at the same time. There were multi-scale features used to give the network global and local information. Their accuracy on KinFaceW-I was 85.6% and on KinFaceW-II it was 88.8%.

13.Zekrini et al.[68] proposed a technique for performing reliable kinship verification based on the combination of two descriptors in 2022. The Gradient Local Binary Patterns (GLBP) is first descriptor, which links gradient and textural information. The Histogram of Templates (HOT) is a shape descriptor. These features are used to define kinship ties in face photos, and SVM is employed to classify kinship, their verification accuracy was 76.99 on the KinfaceW-II and 90.49 on the Cornell datasets, as shown in Table 2.2. In this thesis, a deep learning technique is used.

Table 2.2: The Related Works Summary.

Authors	Year	Method	Dataset	Accuracy
Wu et al. [56]	2019	SMCNN	KinFaceW-I	72.7 %
			KinFaceW-II	79.25 %
Chergui et al. [57]	2019	(ResNet) for the feature extraction +SVM	Cornell KinFace	87.16 %
			UB KinFace	83.68%
			Family 101	82.07%
			KinFaceW-I	79.76%
			KinFaceW-II	76.89 %
Chergui et al. [58]	2019	(LBP, LPQ, BSIF) and the Multi-Block (MB) representation + SVM	Cornell KinFace	84.74 %
			UB KinFace	82.89 % 81.69 %

			KinFaceW-I KinFaceW-II Family 101	80.12 % 78.16 %
Nandy and Mondal [59]	2019	Using Siamese CNN. it consists of two parallel SqueezeNet Networks + cosine similarity.	FIW dataset	67.66%
Van & Hoang [60]	2019	Use (LBP) for the feature extraction,Then the features are computed by (Chi-Square) then use feature selection + SVM.	KinFaceW-I KinFaceW-II	72.6% 81.8%
Zhang et al. [61]	2019	Use (CNN) for deep appearance and shape feature extraction.	KinFaceW-I	78.3%
Goyal and Meenpal [62]	2019	Use (HOG, LBP) for the feature extraction + SVM	KinFaceW-I	75.57%
Mukherjee & Meenpal [63]	2019	Use (CLBP) and (LFDA) + KNN classifier.	KinFaceW-I KinFaceW-II	82.82% 89.36%
Crispim et al. [64]	2020	Collected the first 3D kinship database. Then, for classification, we use CNN and SVM.	KinFaceW-I KinFaceW-II Kin3D	75.85% 85.6% 90%
Rachmadi et al. [65]	2020	Uses (FA-CNN) classifier constructed based on FaceNet CNN architecture	FIW	68.84%
Wu et al. [66]	2021	Use SIFT, then utilize two different techniques, feature combination and component-based metric learning (CML)	KinFaceW-I ,KinFaceW-II	73.8% 78.23%

		techniques, to verify kinship.		
Yan & Song [67]	2021	Deep relational network.	KinFaceW-I KinFaceW-II	85.6% 88.8%
Zekrini et al. [68]	2022	Combined features are the (GLBP), (HOT)+ SVM	Kinface W-II Cornell datasets	76.99 % 90.49 %

CHAPTER THREE

PROPOSED METHODOLOGY

3.1 Overview

This chapter presents the architecture of the proposed system to verifying the facial kinship by employing three Dimensional Convolutional Neural Networks (3D CNN). The proposed model consists of two main stages: the preprocessing stage then kinship verification stage, and each stage include multiple steps, which perform different functions.

There are additional processes implemented on the dataset, which is an evaluation process implemented to evaluate the performance of the proposed system depend on test data.

3.2 Kinship Verification System Deployment Stages:

The main stages of building and using a proposed model, which can be seen in Figure 3.1, are:

3.2.1 Data splitting (dataset division):

Preparing and dividing a dataset is an essential first stage in the machine learning model. It involves organizing and splitting the data into appropriate subsets to facilitate training, validation, and testing of models. Each dataset is divided into three subsets: training set, validation set, and test set. The training set is utilized to train the model, the validation set is utilized to modify hyperparameters and monitor the performance of model, and the test set is utilized to evaluate the final performance of model.

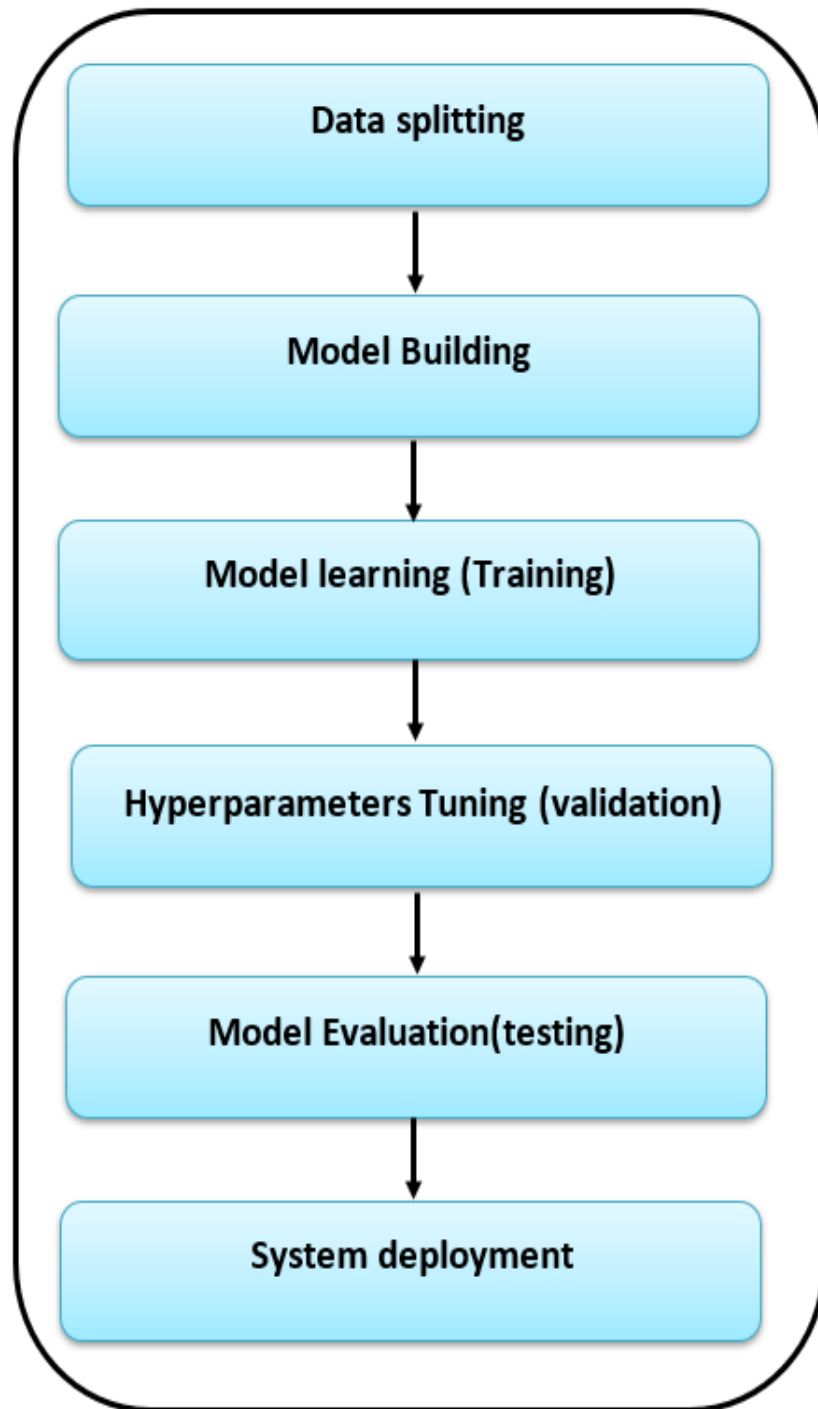


Figure 3.1: Main Stages of The Proposed System

3.2.2 Model Building:

Building an appropriate deep neural network with stacked suitable layers based on the data's characteristics and specific requirements of the facial kinship verification task. Figure 3.2 illustrates the Architecture of the Proposed 3D CNN.

3.2.3 Model Training:

In this stage, the built model is trained on the training set. The model learns from the two facial input images and their corresponding output kinship labels, adjusting its internal parameters to minimize the prediction error.

3.2.4 Hyperparameters Tuning:

3D CNN such as other deep neural network models have hyperparameters that control the learning process. Hyperparameters tuning involves finding the best combination of hyperparameters values that yield the optimal model performance on the validation set. Tuning the deep neural network Hyperparameter is called a dark art, so there are not find specific manners to choose and adjust them; therefore, the trial and error way is used.

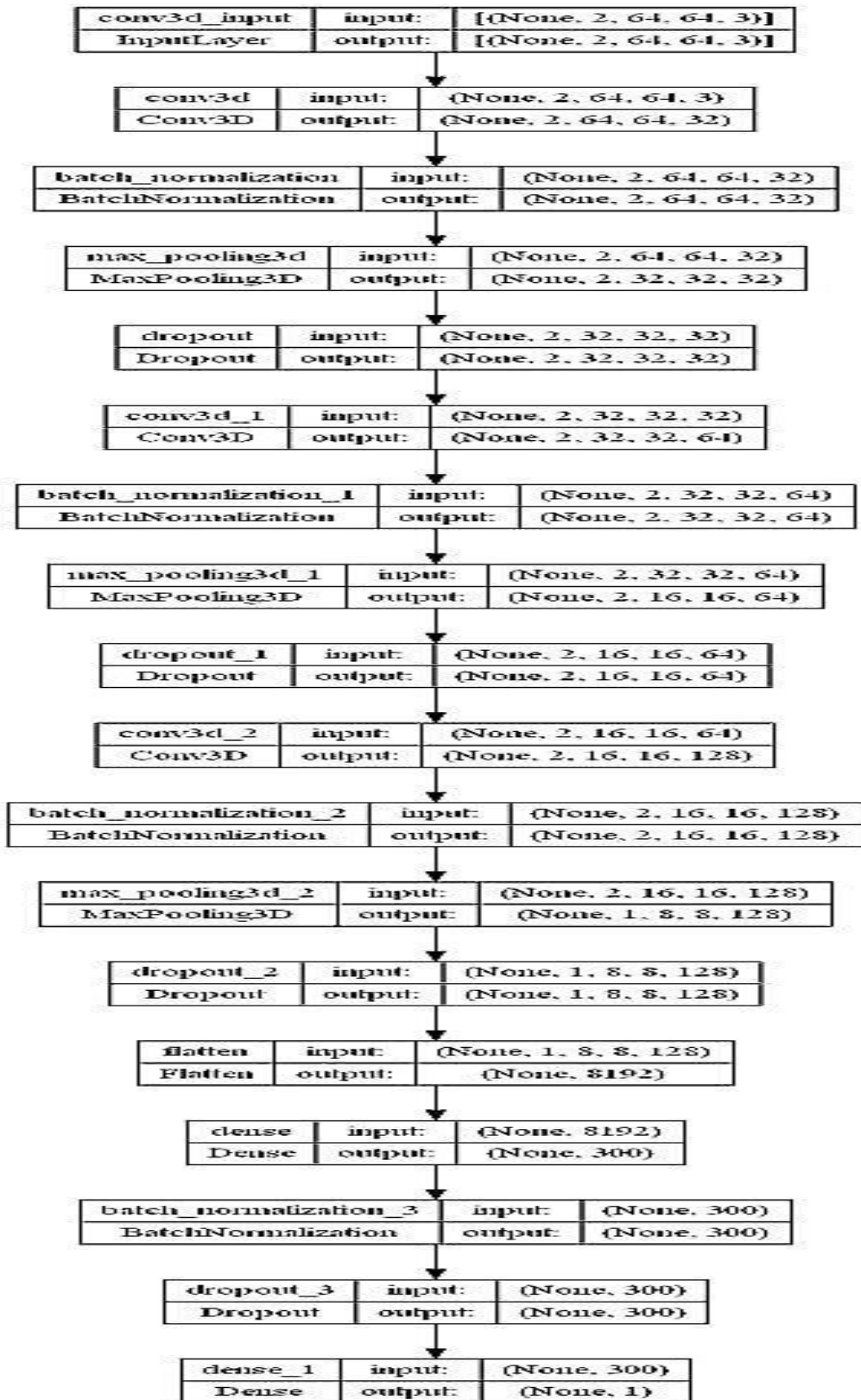


Figure 3.2: Architecture of the Proposed 3D CNN

3.2.5 Model Evaluation (testing):

After training and tuning the model's hyperparameters, the model is evaluated on the test set. The model's performance is evaluated using a variety of evaluation metrics, including accuracy, recall, precision, F1-score, confusion matrix, and K-fold cross-validation.

3.2.6 System Deployment:

Once the model demonstrates satisfactory performance on the test set, it can be deployed in real-world applications to make testing on new, unseen data. During the deployment phase, the system can be integrated into a larger system or application.

3.3 Proposed Model Architecture

The proposed model is built in two major stages: the preprocessing stage and the kinship verification Stage. Each stage has many inner steps, as shown in Figure 3.3, and Algorithm (3.1) describes these work stages.

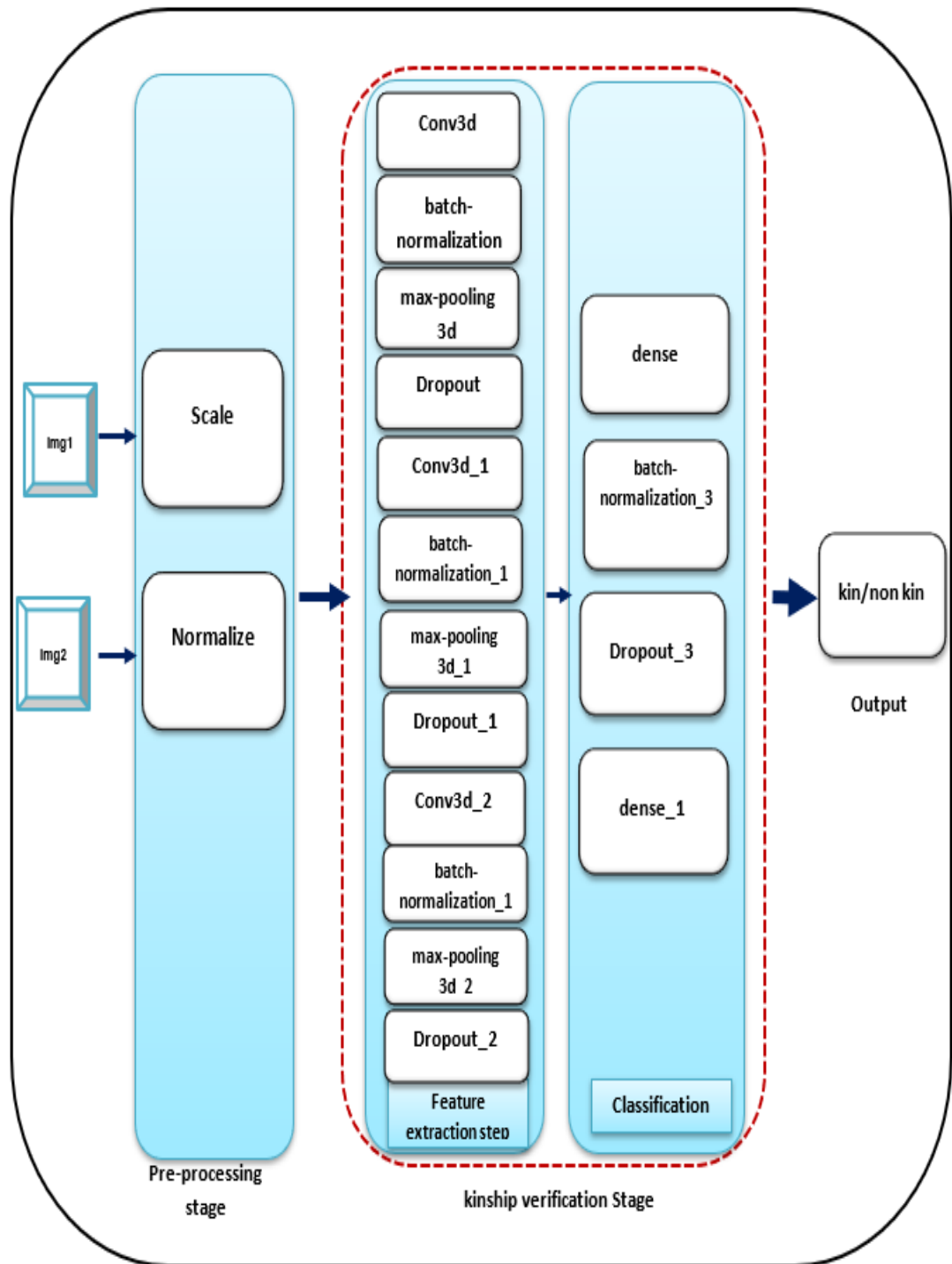


Figure 3.3: Block Diagram of The Proposed Model

Algorithm (3.1): The Proposed Work Stages

Input:

Input the number of samples, each sample contain two images.

Output:

3D CNN model: kinship verification model, the decision for the kinship of two images is kin or not.

Begin

Step 1: Data Preparation

Gather dataset of facial images labeled with kinship information preprocess the facial images (scale by applying equation (8), and normalization by applying equation (10)) and split the dataset into Training dataset and Testing Dataset.

Step 2: 3D CNN model architecture

- a. Input: (pair facial image).
- b. Process.
- c. Output: (value close to 1 refer to kin, 0 refer to non-kin).

Step 3: Training the model and saved weights.

Step 4: Test the model 3D CNN and return the predicted class.

Step 5: Evaluation stage by applying equations (11), (12), (13), (14).

End.

3.3.1 Pre-processing stage

In this stage prepares the input image to implemented on the proposed system by scaling and normalizing. This stage enables the generalization of the system for handle any input image.

- a) **Resize (scale):** Image scaling means resizing an image, which involves rebuilding it from one pixel grid to another, this is done in this step by decreasing or increasing the sum of all the pixels contained in an image, which brings it to (64X64). It is a more important step to make sure that the results are valid for all data and that the deep learning model can use them, it done according to Equations (8), (9) in Chapter 2.
- b) **Normalization:** Image data pixel values are integer numbers between 0 and 255 that represent the intensity of the pixels. When processing inputs, neural network models must deal with small weight values. Large integer values may slow down or disrupt the learning process. Normalization is a process that changes the range of pixel intensity values. Normalization of values is done according to Equation (10) in Chapter 2.

3.3.2 Kinship Verification Stage

Kinship Verification is done by artificial intelligent tools by using a 3D CNN model, the next stage implements a Classification for verifying facial kinship. This stage includes two major steps (feature extraction step and then classification step), both of them is constructed up of multiple layers that perform diverse functions depending on the goal of each layer. Table 3.1 illustrates a summary representation of the proposed system

architecture, which shows information about all layers and their order in the proposed system, the output shape and number of parameters (weights) associated with each layer, Additionally, the table provides the total number of proposed system parameters (weights).

Table 3. 1: The Proposed System

Layer (type)	Output Shape	Param #
conv3d (Conv3D)	(None, 2, 64, 64, 32)	2624
Batch_normalization (Batch Normalization)	(None, 2, 64, 64, 32)	128
max_pooling3d (MaxPooling3D)	(None, 2, 32, 32, 32)	0
dropout (Dropout)	(None, 2, 32, 32, 32)	0
conv3d_1 (Conv3D)	(None, 2, 32, 32, 64)	55360
batch_normalization_1 (Batch Normalization)	(None, 2, 32, 32, 64)	256
max_pooling3d_1 (MaxPooling3D)	(None, 2, 16, 16, 64)	0
dropout_1 (Dropout)	(None, 2, 16, 16, 64)	0
conv3d_2 (Conv3D)	(None, 2, 16, 16, 128)	221312
batch_normalization_2 (Batch Normalization)	(None, 2, 16, 16, 128)	512
max_pooling3d_2 (MaxPooling3D)	(None, 1, 8, 8, 128)	0
dropout_2 (Dropout)	(None, 1, 8, 8, 128)	0
flatten (Flatten)	(None, 8192)	0
dense (Dense)	(None, 300)	2457900
batch_normalization_3 (Batch Normalization)	(None, 300)	1200
dropout_3 (Dropout)	(None, 300)	0
dense_1 (Dense)	(None, 1)	301
Total params: 2,739,593		
Trainable params: 2,738,545		
Non-trainable params: 1,048		

The total parameters obtained from sum of all layers parameters except the parameters of batch normalization layer that is represent nontrainable parameters.

3.3.2.1 Feature Extraction Step

The Feature Extraction step utilizes a three-dimensional Convolutional Neural Network (3D CNN) model for extracting the salient features from the input samples. It is constructed up of multiple layers, including 3D convolution layers, nonlinear layer, pooling layers, batch normalization layers, and dropout layers, that perform diverse functions depending on the goal of each layer. The saved weights are used to compute the output of each layer and fed to the next layer until the decision-making (model's output) in the last layer in order to determine whether the two input images have kinship or not. The feature maps that output from the Feature Extraction step are Flatten before inter to the last step (Classification step/ Kinship Verification Step).

This step consists of three convolution layer each one followed by one batch normalization layer and one subsampling layer and dropout layer. 3D convolution layers employ 3D kernels (filters) on two face images to determine important features then produce feature maps. The number of filters represents the layer's depth and have size (32, 64, 128) respectively for the three convolution layers with kernel size (3, 3, and 3). The kernel coefficient values can be selected through the training process and were represented by the saved weights.

To represent non-linear layers, we use the "Rectified Linear Units (ReLUs) function" on all layers and the "sigmoid function" on the

output layer. These functions are used to find the robust features of all hidden layer.

Batch normalization layers are added to the model to accelerate the training process and coordinate the update of multiple layers. The general process is sketched in Figure 3.4.

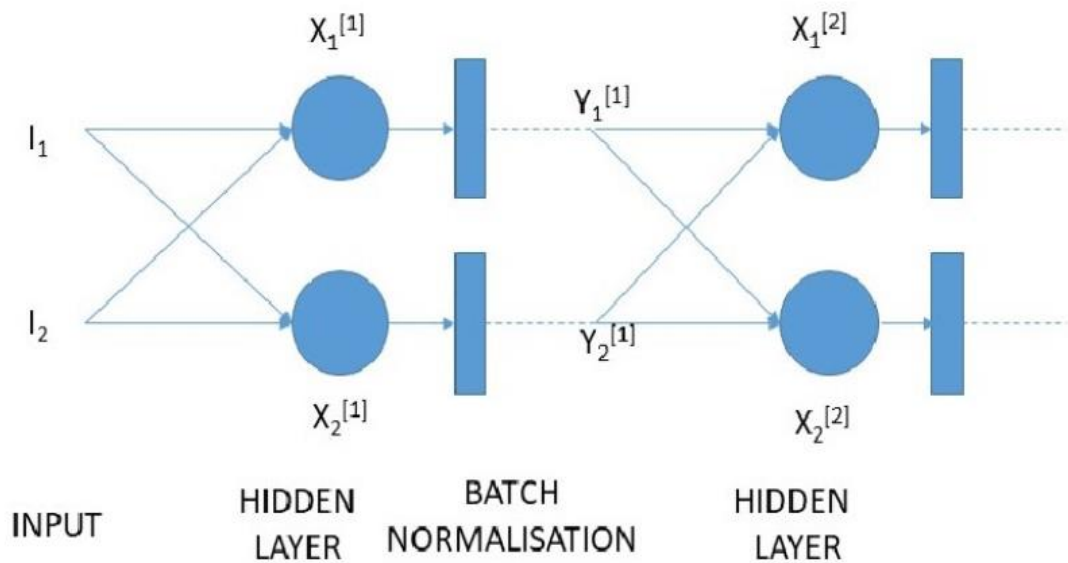


Figure 3.4: Batch Normalization Sketch for Simple Network

Max-pool layer provides resilience against noise by decreasing the resolution of the features by passing a single neuron with maximum value in one layer from the clustered of several neurons in the previous layer using a max-pool function of clustered neurons. These clusters of neurons are made by dividing the input images into three-dimensional spaces that don't overlap. considering each space as a cluster, and selecting the maximum value for each cluster. Figure 3.5 illustrates Max Pooling process.

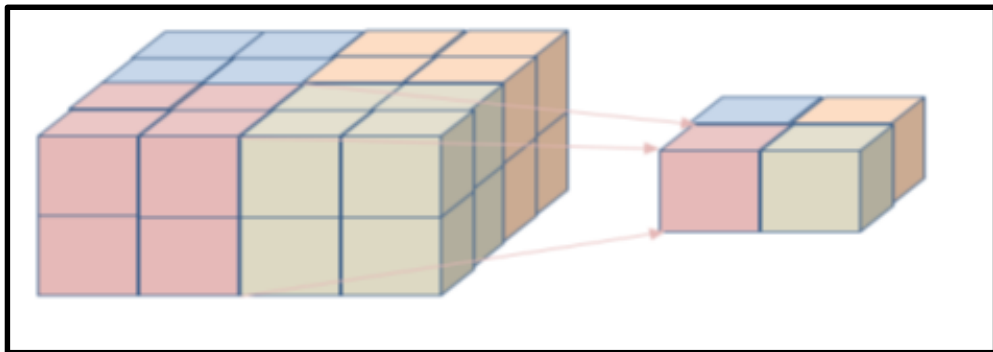


Figure 3.5: The Max Pooling process

3.3.2.2 Classification Step

Flatten layer flatten the output of previous layers and classify a samples of face images. This step is implemented by utilizing a fully connected layer, also known as a Dense layer, consisting of 300 neurons. These neurons employ the "ReLU" activation function. The final layer, known as the output or decision-making layer, is a fully connected layer that employs a sigmoid function to provide the final class. In Classification step, the Batch Normalization layer and Dropout layer also adds and followed FC layers.

Dropout layers are included in the proposed system to avoid overfitting and making generalizations on unseen data. During the training process, it chooses 50% of the neurons at random and sets their weights to zero. It is an easy way to reduce the model's sensitivity to noise while it is being trained, while keeping the required level of complexity for the architecture of the proposed system.

3.4 Model Evaluation

In the stage of model evaluation, the proposed model is evaluated utilizing the test dataset. The models of deep learning are stochastic. This indicates that these models contain variable processes at all times. So, the outcome leads to some randomness and some uncertainty by differing predictions with differing overall skills, despite the fact that the same data was used. Because DNNs have many parameters, they may over-fit the training data throughout the learning process. This means that the model performs excellent with training data but fails when generalizing to unseen data. This results in poor performance on new data (typically the test set). Therefore, the model is evaluated in two skills:

3.4.1 Stochastic Model's Skill Estimation (Model Stability Controlling):

It provides different results when training the same model with the same data and evaluating the non-stochastic model multiple times, then obtaining the mean value. The Log Loss, Confusion Matrix, Accuracy, Precision, Recall, and F1-Measure metrics are utilized to assess the proposed system.

3.4.2 Model Skill Estimation (Model Variance Controlling):

When training the same model with different data, different results are obtained. Cross Validation is used to create more reliable evaluations. The process of training and testing is then repeated ten times, employing a 10-folds cross validation technique.

3.5 K-Folds Cross Validation

The cross-validation process measures a machine learning model's skill on unseen data by evaluating models on specified resampling data sets depending on a single parameter called k .

The proposed model is given $k = 10$, for each value of K , it split the dataset into Training (80%) + Validation (10%) = 90% and Testing = 10% it run the algorithm (3.2), recording the performance of testing based on the metric employed (adopted accuracy). Finally, the result is represented by the average of the performance.

Algorithm (3.2): 10-Fold Cross-Validation

Input: Total Dataset

Output: final Testing accuracy.

Step 1: Shuffle the dataset randomly

Step 2: Determine the value of K as K is 10, Split the dataset into k groups.

Step 2: for each value of K :

a) Select unique training and testing datasets by take the group as testing set and the remaining groups as a training data set:

$KFoldTesting = \text{subset}(\text{Data})$

$KFoldTraining = \text{subset}(\text{Data})$

b) Train and record performance

$KFoldPerformance[i] = \text{Train}(KFoldTraining, KFoldTesting)$

c) Retain the evaluation score and discard the model

Step 3: Summarize the performance skill of the model

$\text{TotalPerformance} = \text{ComputePerformance}(KFoldPerformance)$

End.

K-Folds Cross Validation provide a more reliable method for evaluating the performance of a model by decreasing the risk of overfitting and variance issues compared to using a single train-test split.

3.6 Models Implementation

In the context of machine learning and AI models, there are two distinct phases that a model can be in during its operation. These are the learning mode and the testing mode. In many real-world scenarios, models are trained offline in learning mode and then deployed to make real-time predictions in testing mode, allowing for efficient and effective decision-making processes in various applications.

- **The learning mode:** Learning mode refers to the phase in which a machine learning model is being trained on a dataset to learn from the patterns and relationships that exist within the dataset to enable the model to produce accurate predictions on new, unseen data by learning from the training examples.

This mode includes the training and validation process. During this phase, the model is exposed to labelled data (data with known inputs and corresponding outputs) and adjusts its internal parameters and weights to reduce the error or differences between its predictions and actual outputs. The weights are continuously updated until the network converges to the minimum error using a loss function.

Following network stability, the validation process is implemented to validate the model on given validation data with

corresponding labels (which is regarded as a complemented part of the learning phase). The preserved training weights are used to prove the performance and accuracy of the trained model and to determine the model's ability to predict classification with new data.

- **Testing mode:** This mode (also known as deployment mode) is the phase in which the trained machine learning model is put into practical use to generate classifications on unseen, new data. Once the proposed model has been trained in the learning mode, it is now ready to be deployed in real-world applications to perform the tasks it was designed for.

During testing mode, the model takes input data and produces an output based on the patterns it learned during the learning phase. The model's performance in this mode is evaluated based on how well it generalizes to new, previously unseen data, as this is the ultimate test of its effectiveness and reliability.

CHAPTER FOUR

RESULTS AND DISCUSSION

4.1 Overview

This chapter evaluates the implementation results after testing the effectiveness of the proposed system from the previous chapter using various parameter settings. It is important to note that each stage of the proposed system is evaluated independently. A public dataset is utilized to ascertain how this model will behave. The next sections show the datasets and system requirements that will be used with the proposed system. The outcomes of the proposed system are described in other sections.

4.2 System Requirement

A computer system that has enough processing capacity is a basic requirement for the software or program that performs machine learning and deep learning on any given dataset. To implement the suggested system, the following will be used:

- **Hardware:**
 1. Central Processing Unit (CPU): Intel(R) Core (TM) i7-12700H (20 CPUs) 2.7GHz.
 2. RAM: Samsung 16 GB.
 3. Graphics Processing Unit (GPU): NVIDIA GeForce RTX 3060 Laptop GPU.
- **Operating System:** Windows 11, 64 bit.
- **Programming Language:** Python

4.3 Dataset

In this thesis, three benchmark datasets are used: the KinFaceW-I dataset, KinFaceW-II dataset (Lu et al., 2014), and FIW dataset (Robinson et al., 2016, 2018).

KinFaceW-I: KinFaceW-I dataset was created by Lu et al.[69]. It was collected over the Internet. The KinFace W-I has 1066 images and 533 pairs, respectively, the kin images in KinFaceW-I are cropped from various photographs, see Figure 4.1. The dataset obtained from the website:

<https://www.kinfacew.com/dataset/KinFaceW-I.zip>

Dataset contains two folders; the first folder consists of the parents-children's images. and these images are arranged in four kinship relations: Father-Son (F-S), Father-Daughter (F-D), Mother-Son (M-S), and Mother-Daughter (M-D). The second folder is the metadata of these images and contains four files: father-dau, father-son, mother-dau, and mother-son, representing four different kinship relations: (F-D), (F-S), (M-D), and (M-S).

This database was collected from the web and captured under uncontrolled environments in terms of gestures, lighting, backgrounds, and expressions. This database contains four kin relations, 156 (F-S), 134 (F-D), 116 (M-S), and 127 (M-D) kinship pairs.



Figure 4.1 Part of The Kinship Pairs in KinFaceW-I dataset [69]

KinFaceW-II: KinFaceW-II dataset was created by Lu et al.[69]. It was collected over the Internet. The images in KinFaceW-II are cropped from the same photograph, Figure 4.2. KinFace W-II has 2000 images and 1000 pairs. The dataset obtained from the website: <https://www.kinfacew.com/dataset/KinFaceW-II.zip>

It contains two folders; the first folder consists of the parents-children's images. And the second folder is the metadata for these images. The KinFaceW-II dataset contains four kin relations

(F–S), (F–D), (M– S), and (M–D), 250 pairs of kinship images for each kin relation.



Figure 4.2: KinFaceW-II Dataset [69]

- 1) **FIW Dataset:** It contains a file for facial images, as shown in Figure 4.3, and a CSV file. The CSV file contains 3 columns: the first column is the path to face for subject 1, the second column is the path to face for subject 2, and the third column contains the label (1/0 means KIN or NON-KIN).

The FIW dataset [70], [71] is by far the largest and most complete kinship dataset for verifying kinship from facial images. FIW dataset is organized by the family tree structure. It includes several images of each family member from various time periods. FIW has first-, second-, and same-generation relatives. FIW consists of 13,000 natural family photographs of 1,000 families. The image pairs distributed across 11 relationships, as seen in Figure 4.3. The dataset obtained from the website:

<https://www.kaggle.com/datasets/anasusman/faces-in-the-wild-fiw?resource=download>

FIW dataset is divided into three categories, the relationships include the following:

- a) There are four types of parent-child groups: father-daughter (F-D), father-son (F-S), mother-daughter (M-D), and mother-son (M-S).
- b) Three types of sibling groups: sister-sister (S-S), brother-brother (B-B), and brother-sister (SIBS).
- c) Four types of grandparent-grandchild groups: grandfather-grandson (GF-GS), grandfather-granddaughter (GF-GD), grandmother-grandson (GM-GS), grandmother-granddaughter (GM-GD).

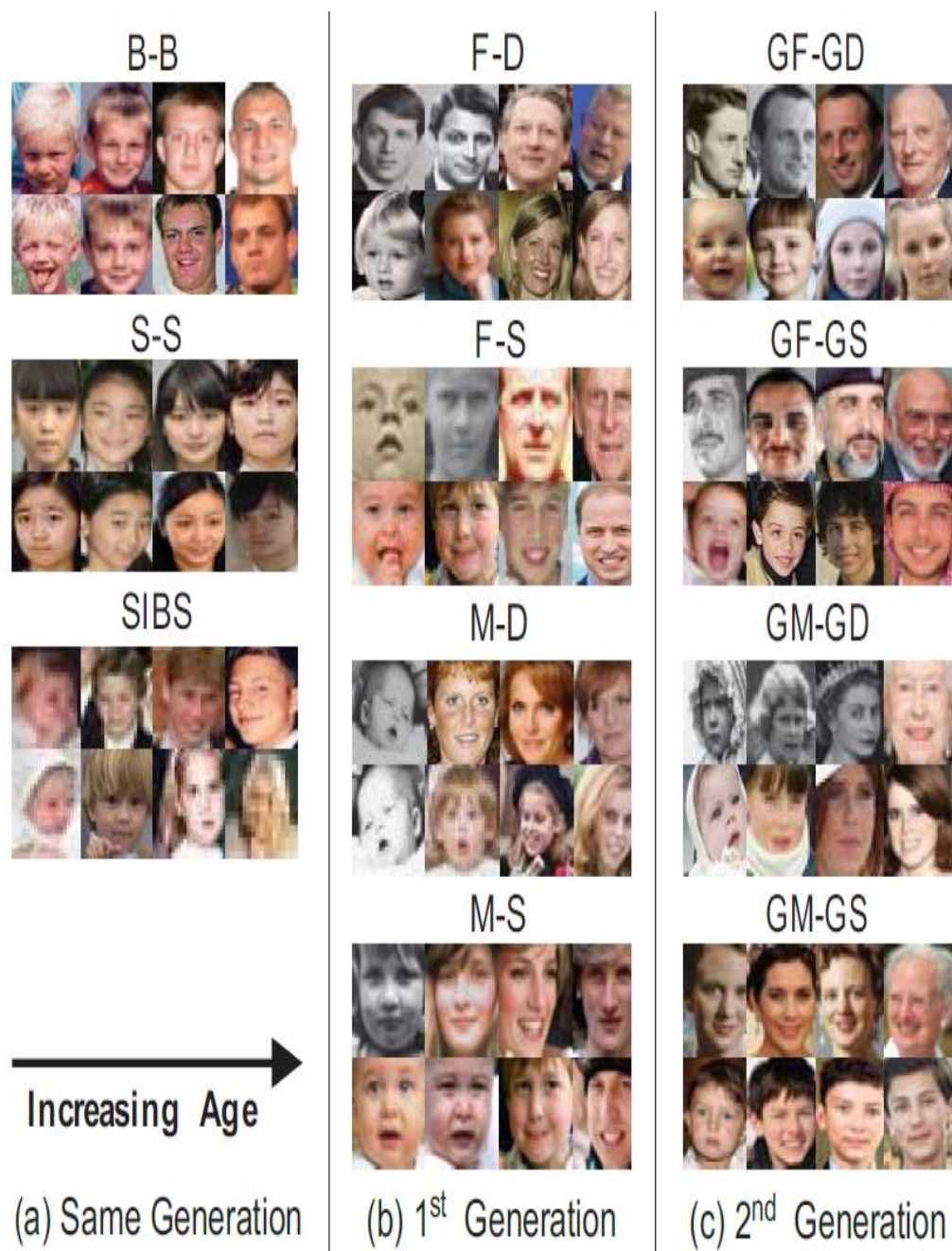


Figure 4.3: Example Face Pairs of Each of The 11-Relationship Type In The FIW Dataset [71].

4.4 Kinship Verification System Deployment stages

The Kinship Verification model include several stages:

4.4.1 Data splitting (dataset division):

The proposed system is implemented on three datasets: the KinFaceW-I dataset, the KinFaceW-II dataset, and the FIW dataset. Each one is separated into two sub-datasets: 85% of the training and 15% of the testing datasets and then the training dataset also is split into 85% for the real training set and 15% for the validation set as seen in Figure 4.4.

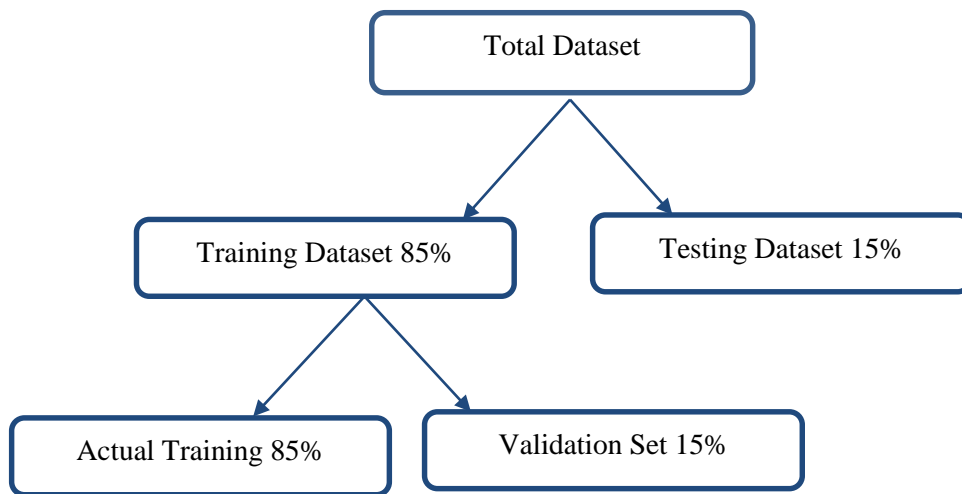


Figure 4.4: The Data Set Division

4.4.2 Model Building:

Many experiments have been implemented to get the best result by changing some parameters such as the number of epochs, learning rate, and batch size. Additionally, adding or deleting layers. The 3D CNN model was built to verify kinship. The proposed model consists of Conv3D layers, Batch Normalization, MaxPooling3D layers, Dropout layers, and fully connected layers.

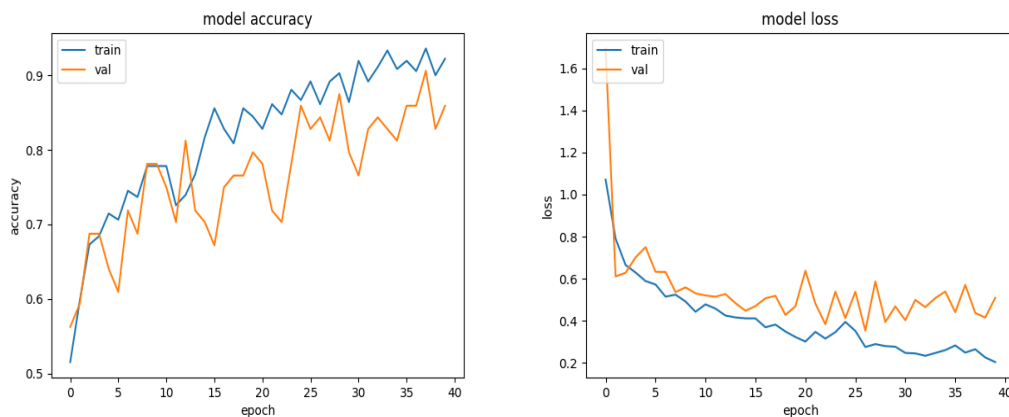
4.4.3 Model Training:

The loss, accuracy, and mean square error functions are employed as metrics in the proposed model. Throughout the training and validation processes, these functions are employed. The proposed model is implemented in two phases: training stage and then testing stage. In the first phase the proposed model is trained using all of the available training data. During the training phase, each layer's weights are updated until the network converges on the lowest error, as measured by the mean square error. Following the model's stability, the validation process is implemented on validation datasets with labels and utilized the kept weights from the training phase to evaluate the performance of model and accuracy and decide if it is ready for testing the label of unseen dataset.

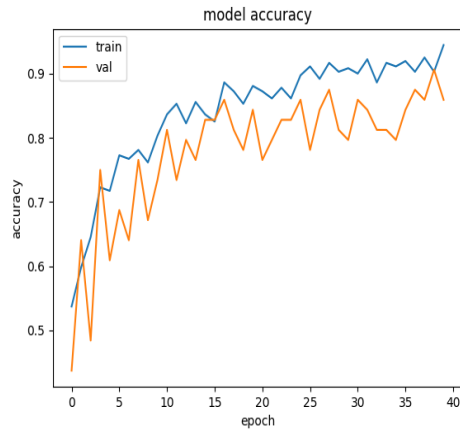
Figure 4.5 illustrates the proposed model's learning behavior on training and validated datasets in all epochs by showing the results of metrics employed in all epochs on the KinFaceW-II dataset, Figure 4.6 for the KinFaceW-I dataset, and Figure 4.7 for the FIW dataset.

- **KinFaceW-II dataset**

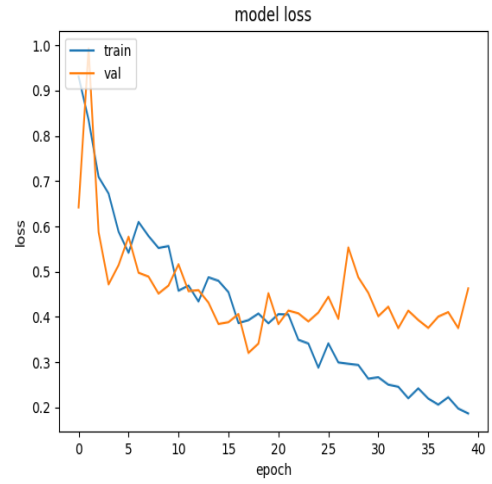
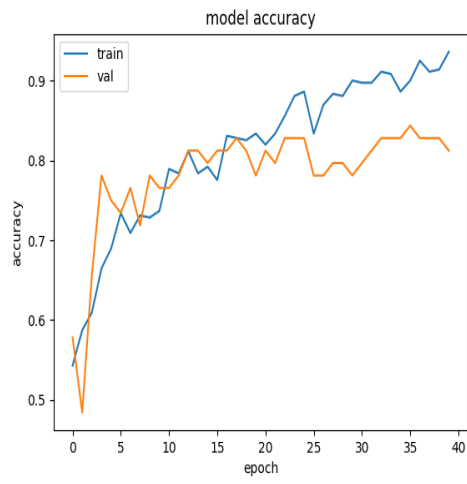
F-S



F-D



M-S



M-D

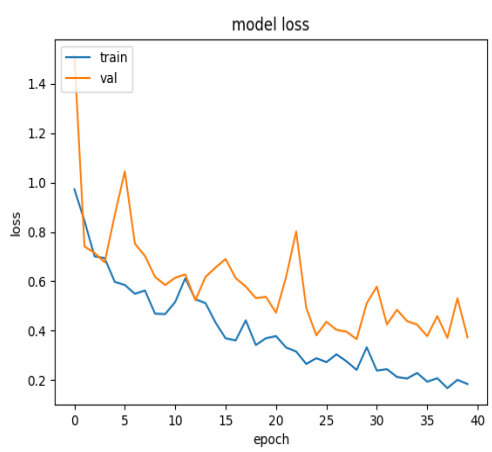
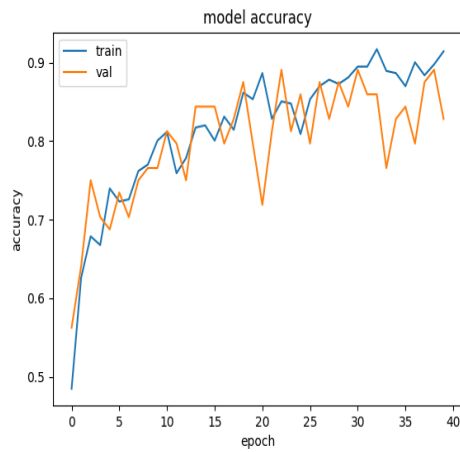
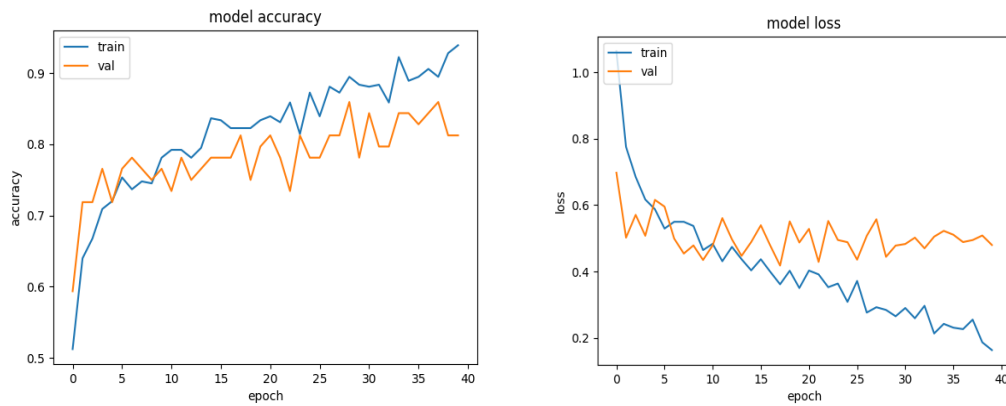


Figure 4.5: Accuracy and Loss Functions of Model KinFaceW-II

By following of the learning behavior of the 3D CNN system during training and validating datasets in 40 epochs as seen in Figure 4.5, for four relations in KinFaceW-II dataset (F-S, F-D, M-S, M-D) we notice the efficiency of the system's performance by increasing accuracy rates and decreasing in loss of function.

- **KinFaceW-I dataset**

F-S



M-D

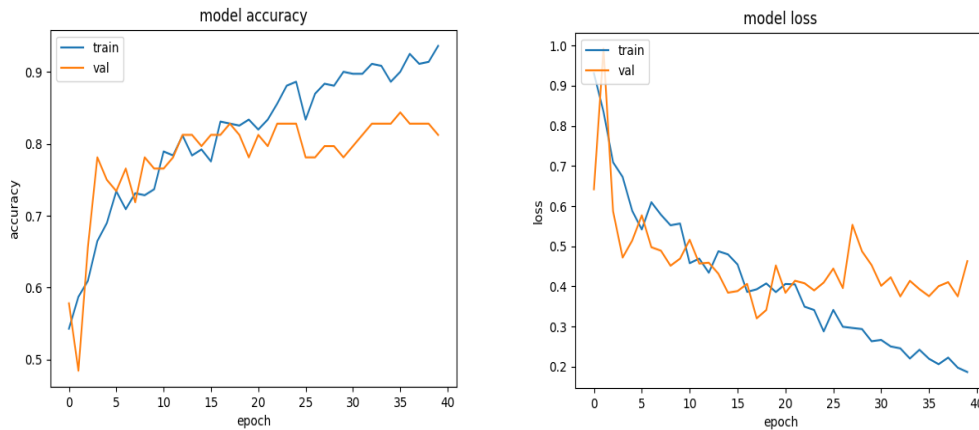
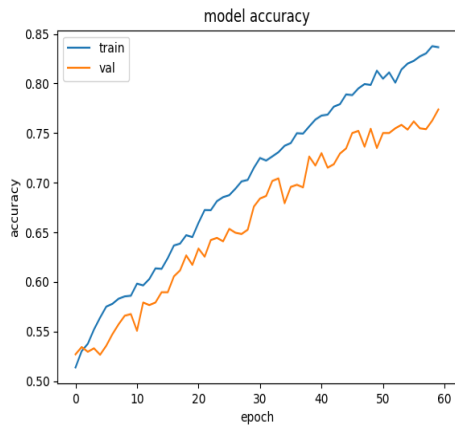


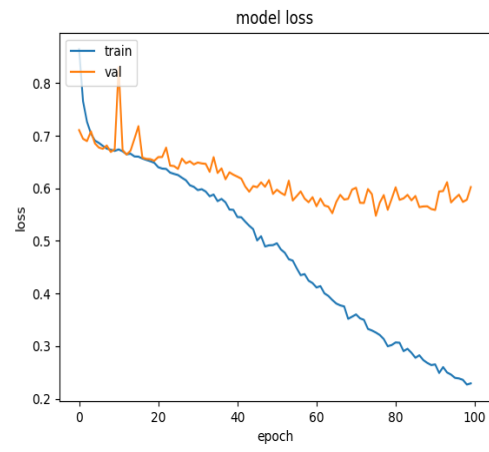
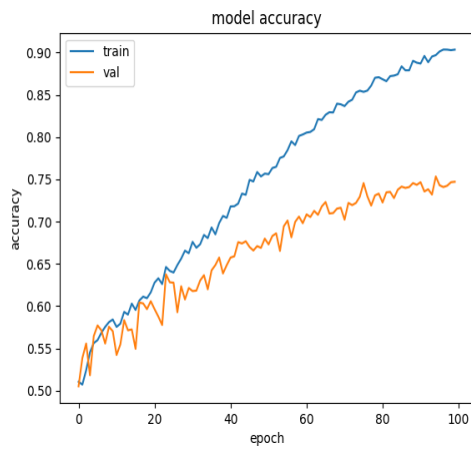
Figure 4.6: Accuracy and Loss Functions of Model KinFaceW -I

- **FIW dataset**

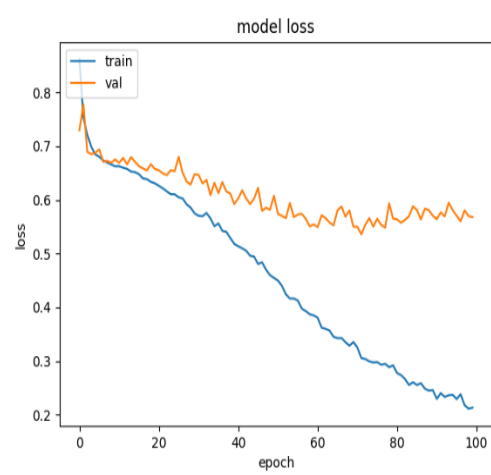
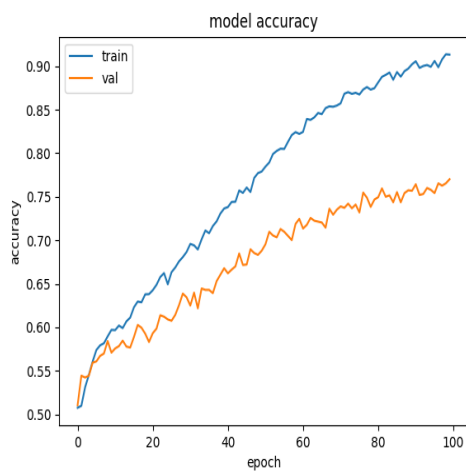
F-S



F-D



M-S



M-D

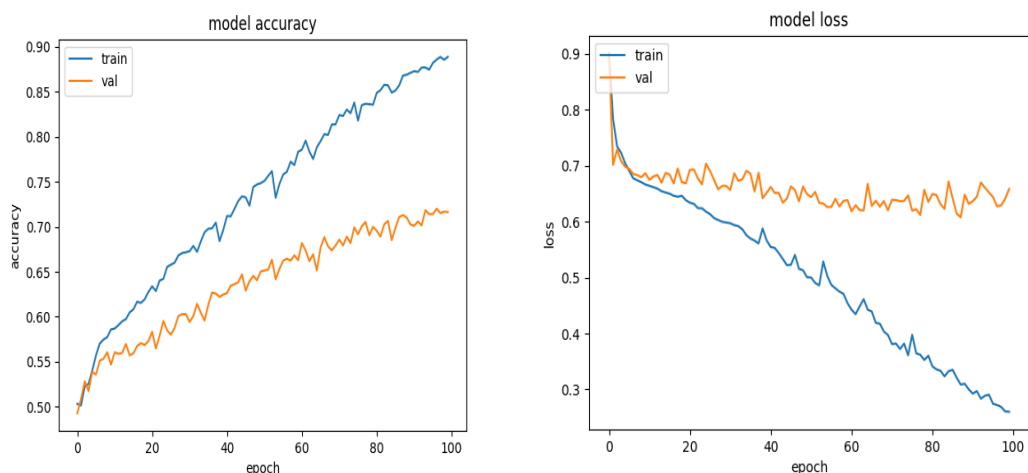


Figure 4.7: Accuracy and Loss Functions of Model FIW

4.4.4 Hyperparameters Tuning:

The hyperparameter constants used to construct our three-dimensional convolutional neural network (3D CNN) model are described in this section. These values were utilized during both the initialization and training stages. The values of the hyper-parameters are displayed in Table 4.1.

Table 4.1: The Utilized 3D CNN Model's Hyper-parameter Values.

Hyper parameter	Value
Learning rate	0.001
Batch size	32
Metrics	Accuracy
Loss function	binary_crossentropy
Optimizer	Adam
Kernel size	3x3x3
Activation function	ReLU and Sigmoid

The results of the experiments relative to the tuning of the model's hyperparameters (batch size, learning rate, and number of epochs) are illustrated in Table 4.2 for the choice of batch size, Table 4.3 for the choice of learning rate, and Table 4.4 for the choice of number of epochs.

Table 4.2: Batch Size Tuning of 3D CNN Model (For Sample M-S KinFace-II dataset)

Batch size	Mode	Loss fun.	MSE	Accuracy
16	Training data	0.122	0.031	0.96
	Testing data	0.516	0.146	0.80
32	Training data	0.188	0.045	0.94
	Testing data	0.414	0.120	0.87
64	Training data	0.326	0.078	0.89
	Testing data	0.556	0.139	0.83

Table 4.3: learning Rate Tuning of 3D CNN Model (for Sample (M-D KinFaceW-I))

learning rate	Mode	Loss fun.	MSE	Accuracy
0.01	Training data	0.240	0.052	0.93
	Testing data	1.056	0.223	0.77
0.001	Training data	0.219	0.051	0.94
	Testing data	0.505	0.146	0.82
0.0001	Training data	0.516	0.174	0.70
	Testing data	0.530	0.184	0.69

Table 4.4: The Proposed Model Experimental Results in Training and Testing Set (For Sample (M-D KinFaceW-I))

No. of epochs	Mode	Loss fun.	MSE	Accuracy
10	Training data	0.421	0.126	0.83
	Testing data	0.690	0.227	0.74
20	Training data	0.400	0.119	0.83
	Testing data	0.585	0.200	0.69
30	Training data	0.299	0.088	0.88
	Testing data	0.740	0.251	0.66
40	Training data	0.219	0.051	0.94
	Testing data	0.505	0.146	0.82
50	Training data	0.238	0.054	0.93
	Testing data	0.638	0.164	0.74
60	Training data	0.178	0.032	0.95
	Testing data	0.704	0.200	0.76
70	Training data	0.229	0.045	0.94
	Testing data	1.034	0.209	0.76

The best accuracy of testing was obtained when used learning rate was changed to 0.001 and the optimizer was Adam. When the number of epochs reaches 40 epochs, and the batch size is 32.

4.4.5 Evaluation (testing)

During the evaluation phase, the proposed model is evaluated by utilizing the test data. Using performance metrics including Precision, Accuracy, F1-Measure, Confusion Matrix, and Recall, the proposed model is assessed, for the KinFaceW-II, KinFaceW-I, and FIW dataset.

1-KinFaceW-II

Table 4.5 displays the facial kinship verification system's performance metrics, while Table 4.6 displays the confusion matrix.

Table 4.5: System's Performance Metrics for Kinship Verification on KinFace-II.

Data (Kinship)	Accuracy	Recall	Precision	F1-score
F-S	93%	93.5%	93.5%	93.5%
F-D	91%	90.5%	90.5%	91%
M-D	93%	93.5%	93%	93%
M-S	87%	85.5%	87%	86%

Table 4.6: The Confusion Matrix CM

a) F-S

N =75		Predicted model	
		No	Yes
Actual class	No	33	3
	Yes	2	37

b) F-D

N =75		Predicted model	
		No	Yes
Actual class	No	34	2
	Yes	5	34

c) **M-D**

N =75		Predicted model	
		No	Yes
Actual class	No	40	3
	Yes	2	30

d) **M-S**

N =75		Predicted model	
		No	Yes
Actual class	No	40	3
	Yes	7	25

These metrics give a comprehensive understanding of how well the model is performing in both positive and negative prediction tasks. In this case, the model seems to perform well, with relatively high accuracy, precision and recall values, indicating it is making accurate predictions for both classes.

Table 4.7 displays the 10-fold cross-validation implemented with the proposed kinship verification model and the experimental results.

Table 4.7: The 10-fold cross-validation of Proposed System in KinFaceW-II

Dataset	Kinship	Accuracy of each fold										model accuracy
		1	2	3	4	5	6	7	8	9	10	
KinFaceW-II	F-S	76.74%	83.72%	81.39%	93.02%	88.37%	92.85%	97.61%	100.0%	100.0%	100.0%	91.37% (+/- 8.05%)
KinFaceW-II	F-D	74.41%	72.09%	93.02%	83.72%	95.34%	100.0%	95.23%	100.0%	100.0%	100.0%	91.38% (+/- 8.24%)
KinFaceW-II	M-S	74.41%	90.69%	88.37%	88.37%	93.02%	90.47%	100.0%	97.61%	95.23%	100.0%	91.82% (+/- 7.14%)
KinFaceW-II	M-D	72.76%	73.74%	90.69%	93.02%	90.69%	95.23%	100.0%	100.0%	95.23%	100.0%	91.14% (+/- 8.68%)

2-KinFaceW-I

Table 4.8 shows the system's performance metrics for kinship verification, while Table 4.9 shows the confusion matrix.

Table 4.8: The Evaluation Measures Values on KinFaceW-I

Data (Kinship)	Accuracy	Recall	Precision	F1-score
F-S	85%	85%	84.5%	84.5%
F-D	85%	85.5%	85.5%	85.5%
M-S	83%	82%	83.5%	82.5%
M-D	82%	82.5%	84%	82%

Table 4.9: The Confusion Matrix CM

a) F-S

N=47		Predicted model	
		No	Yes
Actual class	No	24	4
	Yes	3	16

b) F-D

N=41		Predicted model	
		No	Yes
Actual class	No	18	4
	Yes	2	17

c) M-S

N=35		Predicted model	
		No	Yes
Actual class	No	17	2
	Yes	4	12

d) M-D

N=39		Predicted model	
		No	Yes
Actual class	No	18	1
	Yes	6	14

The 10-fold cross-validation implemented with the proposed kinship verification model and the experimental results can be represented in Table 4.10.

Table 4.10: The 10-Fold Cross-Validation of Proposed System in KinFaceW-I

Dataset	Kinship	Accuracy of each fold										model accuracy
		1	2	3	4	5	6	7	8	9	10	
KinFaceW-I	F-S	68.75	65.62	77.41	93.54	87.09	93.54	100.0	93.54	93.54	100.0	87.31% (+/- 11.79%)
KinFaceW-I	F-D	70.25%	74.07%	62.96%	66.66%	92.59%	88.88%	95.29%	92.59%	90.0%	100.0%	83.33% (+/- 15.11%)
KinFaceW-I	M-S	70.23%	80.76%	61.53%	87.46%	83.99%	72.00%	100.0%	92.00%	83.99%	95.99%	82.80% (+/- 11.60%)
KinFaceW-I	M-D	65.38	88.46	73.07	96.15	100.0	83.99	95.99	92.00	100.0	92.00	88.71% (+/-10.94%)

3-FIW dataset

Table 4.11 shows the system's performance metrics for facial kinship verification, while Table 4.12 shows the confusion matrix.

Table 4.11: The Evaluation Measures Values on FIW Dataset

Data (Kinship)	Accuracy	Recall	Precision	F1-score
F-S	77%	77.5%	77.5%	77%
F-D	75%	74.5%	74.5%	74.5%
M-S	77%	77%	77.5%	77%
M-D	73%	72.5%	73%	73%

Table 4.12: The Confusion Matrix CM

a) F-S

N=2725		Predicted model	
		No	Yes
Actual class	No	1114	251
	Yes	371	989

b) F-D

N =2256		Predicted model	
		No	Yes
Actual class	No	846	311
	Yes	264	835

c) M-S

N=2221		Predicted model	
		No	Yes
Actual class	No	836	281
	Yes	228	876

d) M-D

N=2160		Predicted model	
		No	Yes
Actual class	No	836	245
	Yes	340	739

4.5 Comparison with State-of-the-Art Models

The comparison with other state-of-the-art methods in KinFaceW-II is illustrated in Table 4.13, KinFaceW-I in Table 4.14, and FIW in Table 4.15.

Table 4.13: A Comparison of The Proposed Approach with Other State-Of-The-Art Methods in Kinfacew-II Dataset.

Author	Method	F-D	F-S	M-D	M-S	Accuracy
Wu et al. [56]	SMCNN	79%	75%	85%	78%	79.25 %
Chergui et al.[57]	ResNet+SVM	76.53%	77.69%	77.13%	76.21%	76.89%
Van & Hoang [60]	(LBP)+SVM	82%	87%	87%	71%	81.8%
Zekrini et al. [68]	(GLBP), (HOT)+ SVM	84.66%	76%	75.33%	78%	76.99 %
The proposed model	3D CNN	91%	93%	93%	87%	91%

Table 4.14: A Comparison of The Proposed Approach with Other State-Of-The-Art Methods in KinFaceW-I

Author	Method	F-S	F-D	M-S	M-D	Accuracy
Chergui et al. [57]	ResNet+SVM	81.11%	79.25%	78.03%	80.65%	79.76%
Goyal and Meenpal [62]	HOG, LBP+SVM	75.6 %	77.8%	73.3%	75.6%	75.57%
Crispim et al. [64]	CNN+SVM	69.2	77.8	72.6	83.8	75.85%
The proposed model	3D CNN	85%	85%	83%	82%	83.75%

Table 4.15: A Comparison of The Proposed Approach with Other State-Of-The-Art Methods In FIW

Author	Method	F-S	F-D	M-S	M-D	Accuracy
Nandy and Mondal [59]	CNN	68.74%	62.53%	67.95%	69.84%	67.265%
Rachmadi et al. [65]	Uses (FA-CNN) model	69.59 %	70.13%	71.90%	72.89%	71.1275%
The proposed model	3D CNN	77%	75%	77%	73%	75.5%

The proposed system's results (3D CNN) are compared with the related works on the same datasets (KinFaceW-I dataset, KinFaceW-II dataset, and FIW dataset) for facial kinship verification, and the results show high accuracy compared with other methods such as SMCNN, CNN model, ResNet, LBP, and the (FA-CNN) model.

Additionally, the proposed model was applied to the seven relations in the FIW dataset, and the result is shown in Table 4.16.

Table 4.16: The Accuracy (%) on The Eleven Relationships of the FIW Dataset

Kinship (FIW)	Accuracy
F-S	77%
F-D	75%
M-S	77%
M-D	73%
B-B	98%
S-S	72%
SIBS	67%
GF-GD	79%
GF-GS	73%
GM-GD	81%
GM-GS	83%

CHAPTER FIVE

CONCLUSION AND FUTURE WORKS

5.1 Conclusion

During the period of this work, there were several notes that can be concluded as outcomes of this study:

1. The proposed system 3D CNN presented in this thesis performs facial kinship verification based on determining efficient feature maps. The employment of deep learning to extract high-level image features from the input samples (two images) by implementing a series of non-linear operations, then classifying these input samples (two images) depending on the extracted feature. Spatial-temporal information of the two image is exploited using a 3D CNN, this temporal information represents the number of images and can be considered to make an efficient decision of kinship verification as well as the proposed system's better accuracy and loss functions than a system based on 2D CNN model. The comparison and outperformance of the 3D CNN system can be seen, which proves the accuracy enhancement with 3D CNN as shown in the experimental results.
2. process more than one image at the same time by using three-dimensional convolutional neural network (3D CNN) lead to detect the facial salient features as well as tracking these features through all input images and extract the related unique facial one in the same model and same time, which theoretically leads to reduce the processing time and complexity.
3. Applying normalization to the dataset leads to increased accuracy of the model.

4. Adding batch Normalization and a Dropout layer leads to reduce the overfitting in 3D CNN model.
5. The ReLU activation function integrated with the (3D CNN) is used to extract salient features and neglect the weak features, therefore can deal with different lighting conditions. The ReLU does that by removing all the black elements from the sequence of frames and keeping only those carrying a positive value.

5.2 Future Work

There are several future directions that can be suggested. These are some of these directions:

1. In the future, the proposed model can be developed to determine the level/degree of kinship of the input images.
2. Map Reduce concept can be used to apply the proposed model in real time.
3. Region Of Interest (ROI) can be used to extract the face image from the image or video, in the real world.

REFERENCES

REFERENCES

- [1] E. Katz, J. Halánek, and S. Bakshi, “Forensic Science - Multidisciplinary Approach,” *Forensic, Leg. Investig. Sci.*, vol. 1, no. 1, pp. 1–3, 2015.
- [2] N. Nader, F. E. Z. El-Gamal, S. El-Sappagh, K. S. Kwak, and M. Elmogy, “Kinship verification and recognition based on handcrafted and deep learning feature-based techniques,” *PeerJ Comput. Sci.*, vol. 7, 2021.
- [3] M. Almuashi, S. Z. Mohd Hashim, D. Mohamad, M. H. Alkawaz, and A. Ali, “Automated kinship verification and identification through human facial images: a survey,” *Multimed. Tools Appl.*, vol. 76, no. 1, pp. 265–307, 2017.
- [4] X. Wu *et al.*, “Facial Kinship Verification: A Comprehensive Review and Outlook,” *Int. J. Comput. Vis.*, vol. 130, no. 6, pp. 1494–1525, 2022, doi: 10.1007/s11263-022-01605-9.
- [5] N. Kohli, “Automatic Kinship Verification in Unconstrained Faces using Deep Learning,” West Virginia University, 2019.
- [6] R. A. H. van Oorschot, B. Szkuta, G. E. Meakin, B. Kokshoorn, and M. Goray, “DNA transfer in forensic science: A review,” *Forensic Sci. Int. Genet.*, vol. 38, pp. 1–77, 2019.
- [7] M. Bordallo Lopez, A. Hadid, E. Boutellaa, J. Goncalves, V. Kostakos, and S. Hosio, “Kinship verification from facial images and videos: human versus machine,” *Machine Vision and Applications*, vol. 29, no. 5, pp. 873–890, 2018. doi: 10.1007/s00138-018-0943-x.
- [8] A. Chergui, S. Ouchtati, S. Mavromatis, S. E. Bekhouche, M. Lashab, and J. Sequeira, “Kinship verification through facial images using CNN-based features,” *Trait. du Signal*, vol. 37, no. 1, pp. 1–8, 2020, doi: 10.18280/ts.370101.
- [9] X. Qin, D. Liu, and D. Wang, “A literature survey on kinship verification through facial images,” *Neurocomputing*, vol. 377, pp. 213–224, 2020, doi: 10.1016/j.neucom.2019.09.089.
- [10] N. Kohli, D. Yadav, M. Vatsa, R. Singh, and A. Noore, “Supervised mixed norm autoencoder for kinship verification in unconstrained videos,” *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1329–1341, 2019, doi: 10.1109/TIP.2018.2840880.
- [11] J. Lu *et al.*, “Kinship verification in the wild: The first kinship verification competition,” *IJCB 2014 - 2014 IEEE/IAPR Int. Jt. Conf. Biometrics*, 2014, doi: 10.1109/BTAS.2014.6996230.

- [12] A. M'charek, "Tentacular Faces: Race and the Return of the Phenotype in Forensic Identification," *Am. Anthropol.*, vol. 122, no. 2, pp. 369–380, 2020, doi: 10.1111/aman.13385.
- [13] W. Jang, A. Chhabra, and A. Prasad, "Enabling multi-user controls in smart home devices," *IoT S P 2017 - Proc. 2017 Work. Internet Things Secur. Privacy, co-located with CCS 2017*, no. January, pp. 49–54, 2017, doi: 10.1145/3139937.3139941.
- [14] J. P. Robinson, M. Shao, and Y. Fu, "To recognize families in the wild: A machine vision tutorial," *MM 2018 - Proc. 2018 ACM Multimed. Conf.*, pp. 2096–2097, 2018, doi: 10.1145/3240508.3241471.
- [15] L. Alzubaidi *et al.*, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *J. Big Data*, vol. 8, no. 1, pp. 1–74, 2021, doi: 10.1186/s40537-021-00444-8.
- [16] M. Usama *et al.*, "Unsupervised Machine Learning for Networking: Techniques, Applications and Research Challenges," *IEEE Access*, vol. 7, pp. 65579–65615, 2019, doi: 10.1109/ACCESS.2019.2916648.
- [17] Mirza Cilimkovic, "Neural Networks and Back Propagation Algorithm," *Inst. Technol. Blanchardstown, Blanchardst. Road North Dublin*, vol. 15, pp. 1–12, 2015.
- [18] R. D. Singh, A. Mittal, and R. K. Bhatia, "3D convolutional neural network for object recognition: a review," *Multimed. Tools Appl.*, vol. 78, no. 12, pp. 1–45, 2019, doi: 10.1007/s11042-018-6912-6.
- [19] C. T. Chen and G. X. Gu, "Generative Deep Neural Networks for Inverse Materials Design Using Backpropagation and Active Learning," *Adv. Sci.*, vol. 7, no. 5, pp. 1–10, 2020, doi: 10.1002/advs.201902607.
- [20] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," *Comput. Sci. Rev.*, vol. 40, p. 100379, 2021, doi: 10.1016/j.cosrev.2021.100379.
- [21] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *Artif. Intell. Rev.*, vol. 53, no. 8, pp. 5455–5516, 2020, doi: 10.1007/s10462-020-09825-6.
- [22] D. A. Roberts, S. Yaida, and B. Hanin, *The Principles of Deep Learning Theory*. Cambridge University Press, 2022.
- [23] P. Goyal, S. Pandey, and K. Jain, *Deep Learning for Natural Language Processing: Creating Neural Networks with Python*. Apress, 2018.
- [24] F. Sultana, A. Sufian, and P. Dutta, "Advancements in image classification

- using convolutional neural network,” *Proc. - 2018 4th IEEE Int. Conf. Res. Comput. Intell. Commun. Networks, ICRCICN 2018*, pp. 122–129, 2018, doi: 10.1109/ICRCICN.2018.8718718.
- [25] S. Pouyanfar *et al.*, “A Survey on Deep Learning: Algorithms, Techniques, and Applications,” *ACM Comput. Surv.*, vol. 51, no. 5, pp. 1–36, 2019, doi: 10.1145/3234150.
- [26] C. C. Aggarwal, *Neural Networks and Deep Learning*. Yorktown Heights, NY, USA: springer, 2018.
- [27] N. D. K. Al-Shakarchy, “Driver Drowsiness Detection Based on Spatio-Temporal Features with 3D Convolutional,” University of Babylon, 2020.
- [28] K. W. Kim, H. G. Hong, G. P. Nam, and K. R. Park, “A study of deep CNN-based classification of open and closed eyes using a visible light camera sensor,” *Sensors (Switzerland)*, vol. 17, no. 7, 2017, doi: 10.3390/s17071534.
- [29] K. O. and R. Nash, “An Introduction to Convolutional Neural Networks,” *Int. J. Res. Appl. Sci. Eng. Technol.*, 2015, doi: 10.22214/ijraset.2022.47789.
- [30] S. P. Singh, L. Wang, S. Gupta, H. Goli, P. Padmanabhan, and B. Gulyás, “3d deep learning on medical images: A review,” *Sensors (Switzerland)*, vol. 20, no. 18, pp. 1–24, 2020, doi: 10.3390/s20185097.
- [31] A. T. M. Ali, “Detecting Vandalism in Crowdsourcing Models,” University of Kerbala, 2023.
- [32] S. Ji, W. Xu, M. Yang, and K. Yu, “3D Convolutional neural networks for human action recognition,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 221–231, 2013, doi: 10.1109/TPAMI.2012.59.
- [33] Q. Lan, Z. Wang, M. Wen, C. Zhang, and Y. Wang, “High Performance Implementation of 3D Convolutional Neural Networks on a GPU,” *Comput. Intell. Neurosci.*, vol. 2017, pp. 1–9, 2017, doi: 10.1155/2017/8348671.
- [34] S. Xu *et al.*, “An Early Diagnosis of Oral Cancer based on Three-Dimensional Convolutional Neural Networks,” *IEEE Access*, vol. 7, pp. 158603–158611, 2019, doi: 10.1109/ACCESS.2019.2950286.
- [35] K. Ovtcharov, O. Ruwase, J. Kim, J. Fowers, K. Strauss, and E. S. Chung, “Accelerating Deep Convolutional Neural Networks Using Specialized Hardware,” *Microsoft Res. Whitepaper*, pp. 1–4, 2015, [Online]. Available: <http://research-srv.microsoft.com/pubs/240715/CNN Whitepaper.pdf>
- [36] V. Dumoulin and F. Visin, “A guide to convolution arithmetic for deep learning,” *MILA, Univ. Montréal*, pp. 1–31, 2018.

- [37] S. R. Dubey, S. K. Singh, and B. B. Chaudhuri, “Activation functions in deep learning: A comprehensive survey and benchmark,” *Neurocomputing*, vol. 503, pp. 92–108, 2022, doi: 10.1016/j.neucom.2022.06.111.
- [38] A. D. Rasamoelina, F. Adjailia, and P. Sincak, “A Review of Activation Function for Artificial Neural Network,” *SAMI 2020 - IEEE 18th World Symp. Appl. Mach. Intell. Informatics, Proc.*, pp. 281–286, 2020, doi: 10.1109/SAMI48414.2020.9108717.
- [39] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *Handb. Approx. Algorithms Metaheuristics*, pp. 1–9, 2012.
- [40] D. C. Cireş, U. Meier, J. Masci, and L. M. Gambardella, “Flexible, High Performance Convolutional Neural Networks for Image Classification,” *Proc. Twenty-Second Int. Jt. Conf. Artif. Intell. Flex.*, pp. 1237–1242, 2013.
- [41] S. Mittal, “A survey of FPGA-based accelerators for convolutional neural networks,” *Neural Comput. Appl.*, vol. 32, no. 4, pp. 1109–1139, 2020, doi: 10.1007/s00521-018-3761-1.
- [42] S. H. S. Basha, S. R. Dubey, V. Pulabaigari, and S. Mukherjee, “Impact of fully connected layers on performance of convolutional neural networks for image classification,” *Neurocomputing*, vol. 378, pp. 112–119, 2020, doi: 10.1016/j.neucom.2019.10.008.
- [43] A. Labach, H. Salehinejad, and S. Valaee, “Survey of Dropout Methods for Deep Neural Networks,” *arXiv:1904.13310v2*, pp. 1–13, 2019.
- [44] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” *Journal. Pract.*, vol. 10, no. 6, pp. 1–9, 2016, doi: 10.1080/17512786.2015.1058180.
- [45] J. Bjorck, C. Gomes, B. Selman, and K. Q. Weinberger, “Understanding Batch Normalization,” *arXiv Prepr.*, vol. 4, no. NeurIPS, pp. 1–12, 2018.
- [46] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, “How Does Batch Normalization Help Optimization?,” *32nd Conf. Neural Inf. Process. Syst. (NeurIPS 2018), Montréal, Canada.*, no. NeurIPS, pp. 1–11, 2019.
- [47] M. H. A. Hashm, “Person Identification Based On Facial Aging Forensics Analysis Using Deep Neural Networks,” University of Kerbala, 2023.
- [48] N. Zhang, J. Luo, and W. Gao, “Research on face detection technology based on MTCNN,” *Proc. - 2020 Int. Conf. Comput. Network, Electron. Autom. ICCNEA 2020*, pp. 154–158, 2020, doi: 10.1109/ICCNEA50255.2020.00040.
- [49] D. Zhou, X. Shen, and W. Dong, “Image zooming using directional cubic

- convolution interpolation,” *IET Image Process.*, vol. 6, no. 6, pp. 627–634, 2012, doi: 10.1049/iet-ipr.2011.0534.
- [50] D. Singh and B. Singh, “Investigating the impact of data normalization on classification performance,” *Appl. Soft Comput.*, vol. 97, p. 105524, 2020, doi: 10.1016/j.asoc.2019.105524.
- [51] S. G. K. Patro and K. K. sahu, “Normalization: A Preprocessing Stage,” *Iarjset*, pp. 20–22, 2015, doi: 10.17148/iarjset.2015.2305.
- [52] M. Kuhn and K. Johnson, *Applied predictive modeling*. springer, 2013.
- [53] T. T. J. Kiran, “Computer Vision Accuracy Analysis with Deep Learning Model Using TensorFlow,” *SSRN Electron. J.*, pp. 319–325, 2020, doi: 10.2139/ssrn.3673214.
- [54] H. Dalianis, “Evaluation Metrics and Evaluation,” in *Clinical Text Mining*, no. 1967, 2018, pp. 45–53. doi: 10.1007/978-3-319-78503-5_6.
- [55] D. Anguita, L. Ghelardoni, A. Ghio, L. Oneto, and S. Ridella, “The ‘K’ in K-fold cross validation,” *ESANN 2012 proceedings, 20th Eur. Symp. Artif. Neural Networks, Comput. Intell. Mach. Learn.*, no. April, pp. 441–446, 2012.
- [56] X. Wu, X. Feng, L. Li, E. Boutellaa, and A. Hadid, “Kinship verification based on deep learning,” in *Deep Learning in Object Detection and Recognition*, 2019, pp. 113–132. doi: 10.1007/978-981-10-5152-4_5.
- [57] Abdelhakim Chergui, Ouchtati, S. Salim Mavromatis, and J. S. Salah Eddine Bekhouche, “Investigating Deep CNNs Models Applied in Kinship Verification through Facial Images,” in *5th International Conference on Frontiers of Signal Processing (ICFSP 2019), Sep 2019, Marseille, France. hal-02400686*, 2019, pp. 82–87.
- [58] A. Chergui, S. Ouchtati, S. Mavromatis, S. Eddine Bekhouche, J. Sequeira, and H. Zerrari, “Kinship Verification using Mixed Descriptors and Multi Block Face Representation,” *Proc. - ICNAS 2019 4th Int. Conf. Netw. Adv. Syst.*, 2019, doi: 10.1109/ICNAS.2019.8807875.
- [59] A. Nandy and S. S. Mondal, “Kinship verification using deep siamese convolutional neural network,” *Proc. - 14th IEEE Int. Conf. Autom. Face Gesture Recognition, FG 2019*, no. August, 2019, doi: 10.1109/FG.2019.8756528.
- [60] T. N. Van and V. T. Hoang, “Kinship Verification based on Local Binary Pattern features coding in different color space,” *2019 26th Int. Conf. Telecommun. ICT 2019*, no. 2, pp. 376–380, 2019, doi: 10.1109/ICT.2019.8798781.

- [61] H. Zhang, X. Wang, and C. C. J. Kuo, “Deep Kinship Verification VIA Appearance-Shape Joint Prediction and Adaptation-Based Approach,” *Proc. - Int. Conf. Image Process. ICIP*, vol. 2019-Sept, pp. 3856–3860, 2019, doi: 10.1109/ICIP.2019.8803647.
- [62] A. Goyal and T. Meenpal, “Kinship verification from facial images using feature descriptors,” in *Advances in Intelligent Systems and Computing*, vol. 768, 2019, pp. 371–380. doi: 10.1007/978-981-13-0617-4_37.
- [63] M. Mukherjee and T. Meenpal, “Kinship verification using Compound Local Binary Pattern and Local Feature Discriminant Analysis,” *2019 10th Int. Conf. Comput. Commun. Netw. Technol. ICCCNT 2019*, no. July, 2019, doi: 10.1109/ICCCNT45670.2019.8944489.
- [64] F. Crispim, T. Vieira, and B. Lima, “Verifying Kinship from RGB-D Face Data,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12002 LNCS, 2020, pp. 215–226. doi: 10.1007/978-3-030-40605-9_19.
- [65] R. Rachmadi, I. Purnama, S. Nugroho, and Y. Suprpto, “Family-Aware Convolutional Neural Network for Image-based Kinship Verification,” *Int. J. Intell. Eng. Syst.*, vol. 13, no. 6, pp. 20–30, 2020, doi: 10.22266/ijies2020.1231.03.
- [66] H. Wu, J. Chen, X. Liu, and J. Hu, “Component-based metric learning for fully automatic kinship verification,” *J. Vis. Commun. Image Represent.*, vol. 79, no. August, p. 103265, 2021, doi: 10.1016/j.jvcir.2021.103265.
- [67] H. Yan and C. Song, “Multi-scale deep relational reasoning for facial kinship verification,” *Pattern Recognit.*, vol. 110, p. 107541, 2021, doi: 10.1016/j.patcog.2020.107541.
- [68] F. Zekrini, H. Nemmour, and Y. Chibani, “Feature Fusion for Kinship Verification Based on Face Image Analysis,” *Lect. Notes Networks Syst.*, vol. 413 LNNS, pp. 486–494, 2022, doi: 10.1007/978-3-030-96311-8_45.
- [69] J. Lu, X. Zhou, Y.-P. Tan, Y. Shang, and J. Zhou, “Neighborhood Repulsed Metric Learning for Kinship Verification,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 2, pp. 331–345, 2014, doi: 10.1109/VCIP.2015.7457930.
- [70] J. P. Robinson, M. Shao, Y. Wu, and Y. Fu, “Families in the Wild (FIW): Large-scale kinship image database and benchmarks,” *MM 2016 - Proc. 2016 ACM Multimed. Conf.*, pp. 242–246, 2016, doi: 10.1145/2964284.2967219.
- [71] J. P. Robinson, M. Shao, Y. Wu, H. Liu, T. Gillis, and Y. Fu, “Visual Kinship Recognition of Families in the Wild,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 11, pp. 2624–2637, 2018, doi: 10.1109/TPAMI.2018.2826549.

الخلاصة

يحتوي الوجه البشري على ثروة من المعلومات التي تتأثر بالوراثة، لذلك غالبا ما يتشارك أفراد الأسرة في سمات وجه مشتركة بسبب قرابتهم. يعد التحقق من القرابة مشكلة صعبة في العديد من التطبيقات المهمة، مثل علم الطب الشرعي، والقياسات الحيوية، والتعرف على الوجوه. يوفر التحقق من القرابة أداة قوية في تطبيقات الطب الشرعي، مما يساهم في حل قضايا الأشخاص المفقودين، وتحليل وسائل التواصل الاجتماعي، وأبحاث الأنساب، والدراسة التاريخية. على الرغم من أن اختبار الحمض النووي هو الوسيلة الأكثر دقة للتحقق من القرابة، إلا أنه للأسف لا يمكن استخدامه في العديد من السيناريوهات، مثل المواقف التي تتطلب معالجة في الوقت الفعلي أو التطبيقات التي يكون لدينا فيها مستخدمون غير متعاونين، كما أنه مكلف أيضا. الهدف الرئيسي من هذه الأطروحة هو التحقق مما إذا كان هناك شخصان لديهما صلة قرابة من خلال تحليل صورتي وجهين معا، واستخراج سمات العلاقة بينهما، ومن ثم تحديد ما إذا كان لديهم قرابة أم لا.

يقدم النظام المقترح نظاما للتحقق من القرابة يعتمد على التعلم التلقائي للميزات التمييزية من صور الوجه باستخدام شبكة عصبية تلافيفية ثلاثية الأبعاد ذات بنية جديدة. يتكون هذا النظام من مرحلتين رئيسيتين: مرحلة المعالجة المسبقة ومرحلة إثبات القرابة، وتتضمن كل مرحلة خطوات متعددة تؤدي وظائف مختلفة. في مرحلة المعالجة المسبقة، يتم إعداد الصور المدخلة لتكون مناسبة لنموذج الشبكة العصبية العميقة عن طريق قياسها وتطبيعها. يتم تنفيذ مرحلة التحقق من القرابة لتوفير قرار القرابة في خطوتين: خطوة استخراج السمات لاستخراج ملامح الوجه البارزة وكذلك تتبع هذه السمات في صورتين مدخلتين، ومن ثم خطوة التصنيف لاتخاذ قرار بشأن تلك الصور: قريب أم لا.

كشفت التجارب المكثفة أن شبكة CNN ثلاثية الأبعاد حققت نتائج واعدة مقارنة بالعديد من الأساليب الحديثة. وصلت دقة النظام المقترح إلى 83.75% في مجموعة بيانات KinFaceW-I، و91% في مجموعة بيانات KinFaceW-II، و75.5% في مجموعة بيانات FIW.



جامعة كربلاء
كلية علوم الحاسوب وتكنولوجيا المعلومات
قسم علوم الحاسوب

التحقق من أدلة قرابة الوجه للمساعدة في تحقيقات الطب الشرعي
بناء على الشبكات العصبية العميقة

رسالة ماجستير
مقدمة الى مجلس كلية علوم الحاسوب وتكنولوجيا المعلومات / جامعة كربلاء وهي جزء من
متطلبات نيل درجة الماجستير في علوم الحاسوب

كتبت بواسطة
رؤى كاظم خلف فرج

بإشراف
أ.م.د. نورضياء الشكرجي