Face Recognition Using Thermal Imaging

Computer Science Graduation Project

Submitted by

Mohamed Amr Hassan El-Sayed

Moaz Salem Abd El-Mawgood Abd El-Mageed

Mostafa Mamdouh Mohamed Ibrahim

Moaaz Hail Nasr El-Deen Ahmed El-Shahed

Abdallah El-Sayed Mohamed Yassien

Mohamed Adel Shaaban Saeed Mohamed

Moussa Abd El-Rasheed Youssef Tawadros

Mohamed Ahmed El-Sayed Abd El-Shafy

Supervised By

Prof. Dr. Alaa Abd El Raheem Ahmed

T.A. Nada Yassine

T.A. Eman Taher

2024

Acknowledgment

First Special thanks to our **families** who always supported us during our academic life, and we could not do it without them.

Second for Our greatest in deep gratitude to **Modern University for Technology and Information (M.T.I)** for the significant role it played in forming our character and personalities, providing us with all means necessary for our educational development, and providing us as well with a wonderful environment to learn in throughout our educational years.

This would not have been possible without **Prof.Dr. Olfat Kamel**, President of the M.T.I University has established such a learning environment, providing the most effective facilities for our job to be done. Thanks to **Prof.Dr.Nabil Deabaas** for his effort. It is a pleasure to thank those who maneuvered this Possible, **Prof.Dr.Mohamed Taher El-Mayah** Dean of the faculty of Computer Science for his unlimited help, support, encouragement, and guidance. **Prof.Dr. Mohamed El-Gazzar** Agent of our faculty.

Prof.Dr. Hanafy Ismail, head of the Computer Science Department. A Special thanks to our supervisor **Prof.Dr. Alaa Abd EL-Raheem Ahmed** we are Honored and got the pleasure of working with as our supervisor and our **T.A Nada Yassin** and **T.A Eman Taher**.

Special thanks to **Prof.Dr. Hesham ElDeeb**, head of the AI Department**, Prof. Dr. Eiman Taha,** head of the Basic Science Department, **Prof.Dr. Hafez Abd**

El-Wahab, head of the Information System Department, **Prof.Dr. Anwar Assal** head of the IT center

As well as the department staff for their effort in teaching us the basics for our computer science degree**, Dr. Marwa Refaie, Prof.Dr. Rasha Mohammad, Dr. Rania Ahmed, Prof.Dr. Elsayed Bakkar and Prof.Dr.Mahmoud El-Shishtawy.**

We are grateful to all the **assistant lectures** who helped us all through our studies with practical demonstrations that helped us to understand and practice what we have learned

Abstract

Face recognition has gained significant attention as a biometric identification method due to its diverse applications in security, surveillance, and access control. In challenging scenarios where visible light is limited or inconsistent, thermal imaging offers a promising alternative by capturing the heat patterns emitted by facial features. This project presents a comprehensive approach to designing a face recognition system using thermal images.

The project begins with an exploration of thermal imaging technology and its relevance to face recognition. A diverse dataset of thermal face images is collected, considering factors such as age, gender, and lighting conditions to ensure robustness and inclusivity. Data preprocessing techniques are applied to enhance the quality of the collected dataset, including noise reduction and normalization.

Feature extraction plays a pivotal role in thermal-based face recognition. Various techniques are investigated, with a focus on adapting established methods to the unique characteristics of thermal images. The selected feature extraction method forms the foundation for subsequent model development.

Convolutional Neural Networks (CNNs) have demonstrated exceptional performance in image recognition tasks.

In this project, a specialized CNN architecture is designed and trained using the preprocessed dataset.

Model training involves parameter optimization and augmentation to improve generalization. The trained model's performance is evaluated using a validation dataset and further refined based on the results.

Real-world scenarios often introduce challenges such as varying lighting conditions and subject positioning. To address these challenges, the developed face recognition model is rigorously tested using a dedicated testing dataset. The system's efficiency in real-time and batch processing is assessed, with a focus on accuracy and Loss metrics.

To ensure practical usability, a user interface is developed to interact with the face recognition system. The trained model is seamlessly integrated into the interface, enabling efficient face recognition in diverse environments.

Table Of Contents

List of Figures:

List of Equations:

List of Tables:

List of Abbreviations:

Abbreviations	Full Name
AI	Artificial Intelligence
CNNs	Convolutional Neural Networks
GUI	Graphical User Interface
ML	Machine Learning
PY	Python
SBD	System Block Diagram
UI	User Interface
UX	User Experience

Chapter One

Introduction

Chapter One Introduction

1.1 Introduction

Face recognition technology is integral to modern advancements, used in areas like security and user authentication. Traditional systems depend on visible light images, but thermal imaging is emerging as a method to enhance accuracy and reliability. This project integrates thermal imaging into face recognition to address issues like varying lighting conditions, aiming to improve overall performance.

Understanding the motivation behind using thermal imaging is crucial. Traditional systems struggle in low-light or insufficient-light conditions. Thermal imaging captures heat emitted by objects, including faces, offering a solution to lighting issues and potentially increasing accuracy in challenging scenarios.

An overview of thermal imaging technology is essential, covering principles like infrared radiation and thermal cameras, and how they differ from visible light cameras. This technical foundation is key for developing a robust face recognition system using thermal images.

Highlighting the advantages of thermal imaging for face recognition is pivotal. Thermal cameras work in diverse lighting conditions, are immune to changes in visible light, and can improve accuracy, especially with partial obscurity or disguise.

Acknowledging challenges is important. Thermal face recognition faces issues like variations in thermal signatures and environmental factors, requiring specialized hardware. Recognizing these helps develop effective mitigation strategies.

Defining project objectives guides development and evaluation. Goals include achieving robust performance in low-light conditions, minimizing false positives, and ensuring compatibility with existing frameworks.

Detailing the methodology ensures transparency and reproducibility. This includes steps like data collection, preprocessing, feature extraction, model training, and evaluation metrics, ensuring scientific soundness and practical feasibility.

1.2 Problem Definition

1.2.1 Cosmetics

- High-resolution security cameras may struggle with accurate facial recognition when individuals wear makeup.
- Visible images might not capture makeup details consistently, impacting the reliability of facial recognition systems.
- Cosmetics can alter facial features, leading to discrepancies in identity verification through security cameras.
- The reliance on visible images might result in false positives or negatives in security protocols.
- Behavioral analysis through facial expressions can be affected by the lack of accuracy in capturing subtle makeup-related nuances.
- Visible images may not provide the necessary details for reliable behavioral analytics in security applications.

1.2.2 Low-lighting

- Low-light conditions compromise the effectiveness of security cameras relying on visible images.
- Shadows, reduced details, and distorted colors can hinder the identification of individuals and objects.
- Security cameras in low-light scenarios might miss crucial details, making it easier for intruders to exploit vulnerabilities.
- The limitations of visible images can impact the overall security of surveillance systems.
- Facial recognition algorithms may struggle to operate accurately in low-light conditions.
- Visible images in such scenarios may not provide sufficient data for reliable facial recognition, impacting security protocols.

1.2.3 Aging and Image

- Biometric recognition through visible images can be affected by the aging process.
- Wrinkles and changes in facial features may introduce challenges in accurately identifying individuals.
- Aging security cameras may experience a decline in image quality, affecting the clarity of visible images.
- This deterioration can impact the overall effectiveness of surveillance systems.
- The aging process may not be accurately represented in visible images, impacting the reliability of incident analysis.
- Security cameras relying solely on visible images may struggle to capture aging-related details critical for investigations.

1.2.4 Authentication Challenges with Photographs

- Optical face recognition systems often struggle to distinguish between a real person and a photograph, posing a significant security risk.
- Complex techniques, such as liveness detection and three-dimensional modeling, are required to accurately verify if a subject is a live person.
- These techniques can involve analyzing subtle cues like eye movements, blinks, and changes in facial expression to determine authenticity.

1.3 Digital Image Processing

Image processing is a method to perform some operations on an image, to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which the input is an image and the output may be an image or characteristics/features associated with that image. Nowadays, image processing is among rapidly growing technologies. It forms core research area within engineering and computer science disciplines too. Image processing basically includes the following three steps:

• Importing the image via image acquisition tools

• Analyzing and manipulating the image

• Output in which result can be altered image or report that is based on image analysis. [2]

1.3.1 Levels of Digital Image Processing

• **Low-level processes:** involve primitive operations such as image preprocessing to reduce noise, contrast, enhancement, and image sharpening. A low-level process is characterized by the fact that both its inputs and outputs are images.

• **Mid-level processing:** on images involves tasks such as segmentation (partitioning an image into regions or objects), it is characterized by the fact that its inputs generally are images, but its outputs are attributes extracted from those images (edges, contours, and the identity of individual objects).

• **Higher-level processing:** involves "making sense” of an ensemble of recognized objects.

1.4 Objectives of the study:

This project explores thermal imaging for accurate and robust face recognition in challenging scenarios, aiming to overcome limitations of visible-light systems such as poor lighting and occlusions.

The primary objectives of this project are:

1.4.1. Investigate the Feasibility and Performance of Thermal Imaging for Face Recognition:

- Evaluate thermal imaging's effectiveness in various conditions and compare it with traditional visible-light methods.

1.4.2 Develop Effective Data Preprocessing and Feature Extraction Techniques:

- Enhance thermal image quality, reduce noise, and develop robust feature extraction methods, including deep learning approaches, for accurate recognition.

1.4.3 Design and Implement Efficient and Secure Thermal Face Recognition Systems:

- Create algorithms that balance accuracy and computational efficiency and explore multimodal authentication by integrating additional biometrics.

1.4.4 Evaluate the System's Performance and Ethical Implications:

- Perform thorough performance evaluations on diverse datasets and analyze potential biases and ethical issues related to privacy, discrimination, and social impact.

1.5 Background

1.5.1 Visible Face Recognition

Objective: Identify and learn unique facial features while minimizing variations due to distance, lighting, pose, and occlusions.

Challenges: Variations in distance, lighting, pose, and occlusions can affect accuracy.

Approaches:

- **Feature-based methods**: Extract key geometric features (e.g., eye distance, nose shape).
- **Appearance-based methods**: Use statistical patterns of pixel intensities.

Face Recognition vs. Face Detection:

- **Face Recognition**: Automatically recognizes or verifies a person by comparing facial features to a database.
- **Face Detection**: Identifies human faces in images or videos but does not recognize individuals.

Functionality: Compares specific facial features (e.g., eye distance, nose length, mouth width) to stored data to identify or verify individuals.

1.5.2 Why Use Thermal Instead of Visible Spectrum for Recognition?

Advantages of Thermal IR spectrum:

- **Heat Measurement**: Detect heat emitted by objects, effective in low/no light conditions.
- **Illumination Invariance**: Less affected by lighting changes, improving consistency.

Challenges with Visible Spectrum:

- **Low Lighting**: Less accurate in poor lighting.
- **Reflectance Dependence**: Varies with lighting conditions.

Thermal IR Benefits:

- **Improved Accuracy**: Better performance under varying lighting conditions.
- **Thermal Characteristics**: Reveals unique thermal properties of the face.
- **Anatomical Information**: Captures consistent, unique features.

Techniques:

- Similar recognition techniques from the visible spectrum are applied to thermal recognition.
- Performance: Appearance-based algorithms perform better with thermal IR images than with visible spectrum images.

1.6 Scope of this study

The scope of this project revolves around exploring the potential of thermal imaging for accurate and reliable face recognition in challenging environments. We aim to investigate its effectiveness under various conditions and develop efficient algorithms for data processing, feature extraction, and system implementation.[5]

1.6.1 Scope:

- **Environmentally robust**: Assessing the ability of thermal imaging to recognize faces under diverse lighting, temperature, and humidity conditions.
- **Algorithm development**: Designing and implementing efficient algorithms for data preprocessing (noise reduction, image enhancement), feature extraction (capturing unique thermal signatures), and recognition (matching extracted features with stored profiles).

- **Real-world considerations**: Developing a system that balances accuracy with computational efficiency and resource constraints, considering potential deployment scenarios.

Propose mitigation strategies and best practices for responsible technology development and deployment.

1.7 Selected Software Tools

#	Name	Usage
1	Microsoft Word 2022	Documentation
2	Microsoft PowerPoint 2022	Presentation
3	Kaggle, Google Collab	CNN Model Construction
4	Figma	UI Design
5	Visual Studio Code	GUI Implementation and Integration

1.8 Documentation Outlines

• **Chapter 1:** (Introduction) includes exploration of thermal imaging for face recognition, emphasizing feasibility, performance, and ethical considerations. It defines digital image processing levels, states objectives including system design and evaluation, and specifies scope. The selected software tools for implementation are also highlighted.

• **Chapter 2**: (Project Planning) Project planning includes project team tasks and distribution among the project group members. Also shows the diagrams made in the planning process like Gantt chart and network diagram.

- **Chapter 3**: (Literature View) This chapter delves into the foundational concepts, key methodologies, and technologies in face recognition, with a particular emphasis on the transition from traditional visible spectrum methods to advanced techniques utilizing thermal imaging and deep learning.

- **Chapter 4**: (System Implementation) in this chapter, we describe the detailed process of implementing our thermal face recognition system, which leverages thermal imaging. The implementation is structured into several key phases, each addressing critical aspects of the system's development.

- **Chapter 5**: (System Testing) This Chapter discusses the testing and evaluation of all previous work; the testing process shows that the system has been tested during and after finishing the system implementation. The evaluation of the system was achieved by the users who used the system.

- **Chapter 6**: (Conclusion and future work) includes the conclusion of the implemented system and the future work that can be performed on the project

Chapter Two

Project Planning

Chapter Two Project Planning

2.1 Introduction

Project management is the practice of orchestrating people, resources, and tasks to achieve a specific goal within a set timeframe and budget. It's essentially the art of turning ideas into reality through meticulous planning, organization, and execution.

Here's a breakdown of the key aspects of project management:

- **Planning and Defining Goals:** This involves clearly outlining the project's objectives, deliverables, and the resources needed to achieve them.
- **Breaking down Work:** Large projects are divided into smaller, more manageable tasks with assigned deadlines and owners.
- **Teamwork and Communication:** Project managers lead and collaborate with teams to ensure everyone is aligned on goals and expectations.
- **Resource Management:** This involves efficiently allocating and utilizing people, equipment, and finances to complete the project.
- **Risk Management:** Identifying and mitigating potential problems that could derail the project is crucial for success.
- **Monitoring and Adapting:** Progress is tracked, and adjustments are made as needed to stay on schedule and within budget.

2.2 Project Planning

Project planning is a crucial process that involves defining project goals, objectives, scope, and timelines to achieve a specific outcome. It is an essential part of project management that lays the foundation for the entire project, ensuring that all project activities are aligned with the project's overall objectives.

Task Name	Duration	Start	Finish	predecessors	Resource Names
• Learning Python • Understand the basics of thermal imaging and its applications in face recognition. • Research existing literature, papers, and technologies related to thermal-based face recognition. • Define the project goals, scope, and requirements.	Oct.2023	10/12	10/31		All students
• Collect a comprehensive dataset of thermal face images	Nov.2023	11/1	11/10	1	Mohamed amr , Moaz hael Mohamed ahmed , Moaz salem
• Ensure diversity in terms of age, gender, ethnicity, and lighting conditionst	Nov.2023	11/1	11/15	1	Abdalla elsayed , Mohamed adel Mousa abdelrashed ,Mostafa mamdouh
• Consider privacy and ethical considerations when collecting and using the dataset	Nov.2023	11/16	11/30	1	All students
•Preprocess the collected dataset, including image resizing, noise reduction, and normalization. • Convert thermal images to a suitable format for further processing • Split the dataset into training, validation, and testing sets.	Dec.2023	12/1	12/31	1	All students
•Explore and choose suitable feature extraction techniques for thermal images. • Popular choices include Local Binary Patterns (LBP), Histogram of Oriented Gradients (HOG), and deep learning-based methods. • Extract features from the training dataset using the chosen technique	Jan.2024	1/1	1/31	2	All students
• Decide on the type of model architecture to use for face recognition. • Convolutional Neural Networks (CNNs) and Siamese networks are commonly used for face recognition tasks. • Research and choose a model that suits the nature of thermal images.	Feb.2024	2/1	2/29	2,3	All students
• initial phase involves assembling a diverse dataset comprising videos or sensor data capturing individuals walking in different scenarios	Mar.2024	3/1	3/15	2,3,4	Mohamed amr , Moaz hael Mohamed ahmed , Moaz salem
• Relevant features are extracted from the data to characterize gait patterns, including step length, step width, cadence, and stride duration	Mar.2024	3/1	3/15	2,3,4	Abdalla elsayed , Mohamed adel Mousa abdelrashed ,Mostafa mamdouh
• Machine learning techniques, such as convolutional Neural Networks (CNNs), are employed to learn and recognize the intricate patterns that differentiate individuals based on their gait.	Mar.2024	3/16	3/31	2,3,4	All students
•System Testing	Apr.2024	4/1	4/30	All process	All students
•System documentation	May.2024	5/1	5/31	All process	All students
•System demo	Jun.2024	6/1	6/12	All process	All students
•System presentation	Jun.2024	6/13	6/30	All process	All students

Figure 2.1 Project Planning

2.3 Gantt chart:

The Gantt chart depicts a timeline for a thermal-based face recognition project. The project appears to be divided into five phases: learning Python, defining the problem, collecting data, preprocessing data, and solving the problem. Currently, the project is in the first phase which is to learn Python.

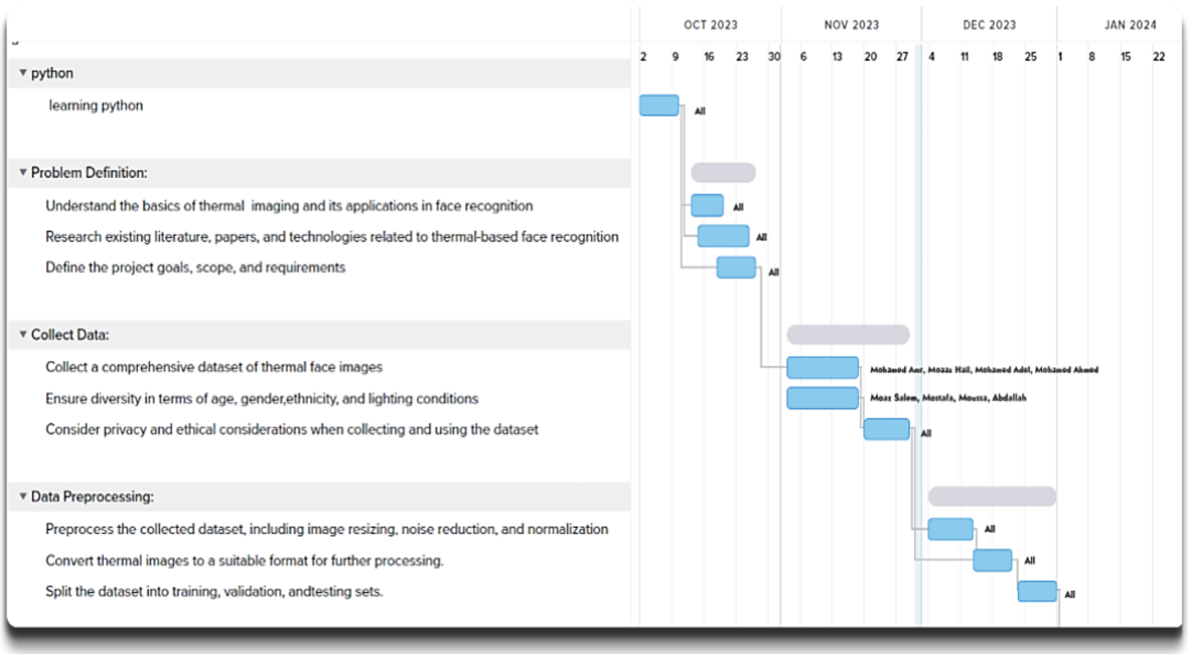

Figure 2.2 Gantt Chart

The thermal-based face recognition project's Gantt chart outlines three phases: Feature Extraction, Model Selection, and System Implementation, the project aims to develop a system for recognizing faces using thermal imaging.

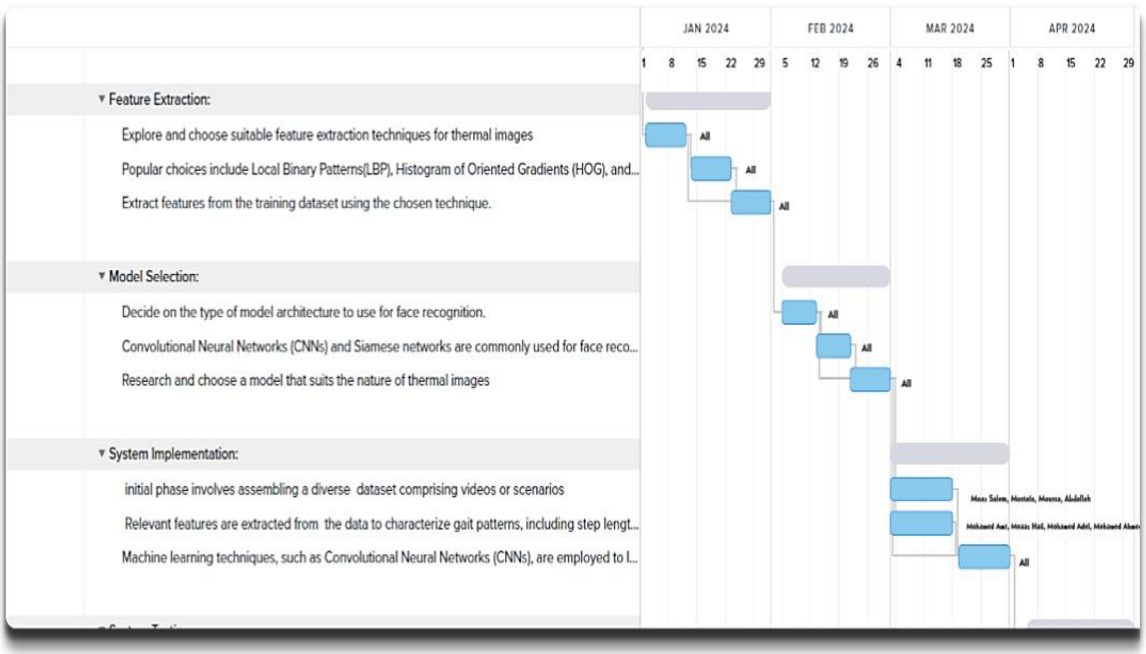

Figure 2.3 Gantt Chart cont.

Testing: Ongoing phase for overall system testing.
Documentation: A completed task for creating system manuals.
Demo: Completed task likely showcasing the system to key users.
Presentation: Upcoming task where the system will be formally presented.

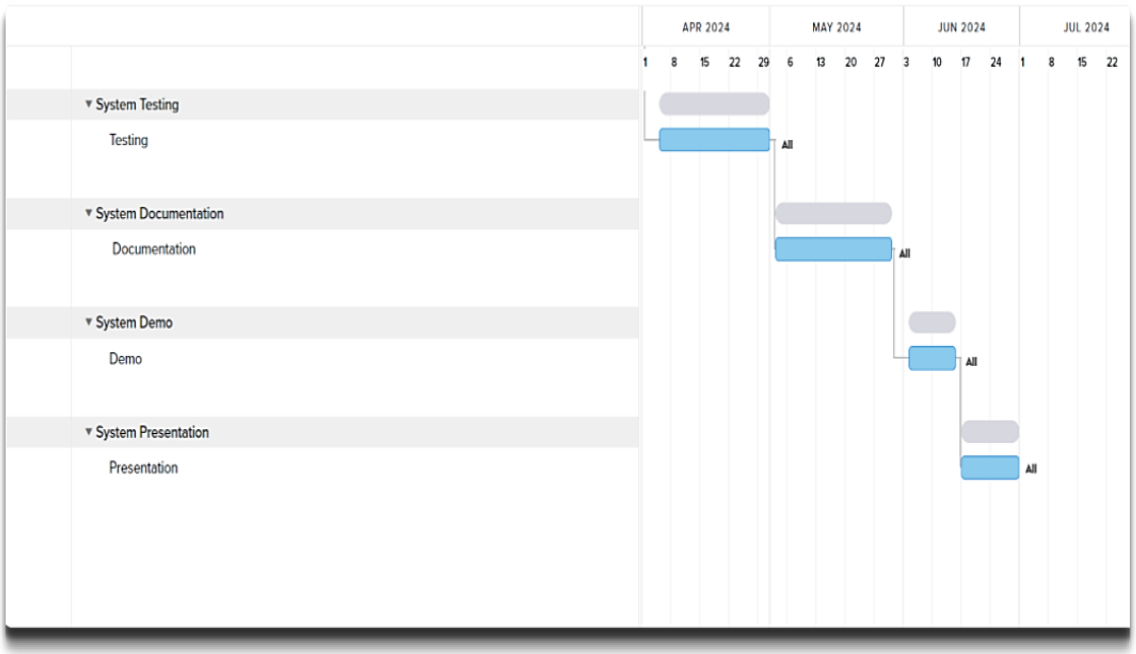

Figure 2.4 Gantt Chart cont.

Chapter Three

Literature Review

Chapter Three Literature Review

3.1 Introduction

In this chapter, we delve into the core concepts, definitions, and cutting-edge technologies pivotal to the field of face recognition. We begin with the fundamental principles and move towards exploring the significant shift from visible spectrum imaging to thermal imaging. The chapter highlights the transformative role of artificial intelligence, particularly focusing on deep learning methodologies. Key topics include the essential activation functions, the architecture and applications of Convolutional Neural Networks (CNNs), and the importance of data augmentation. We also examine Recurrent Neural Networks (RNNs) and conclude with a comprehensive overview of the steps required to develop successful face recognition.

3.2 Definitions

Preprocessing:

As shown in Figure 3.1, which illustrates the angles of the captured preprocessing techniques:

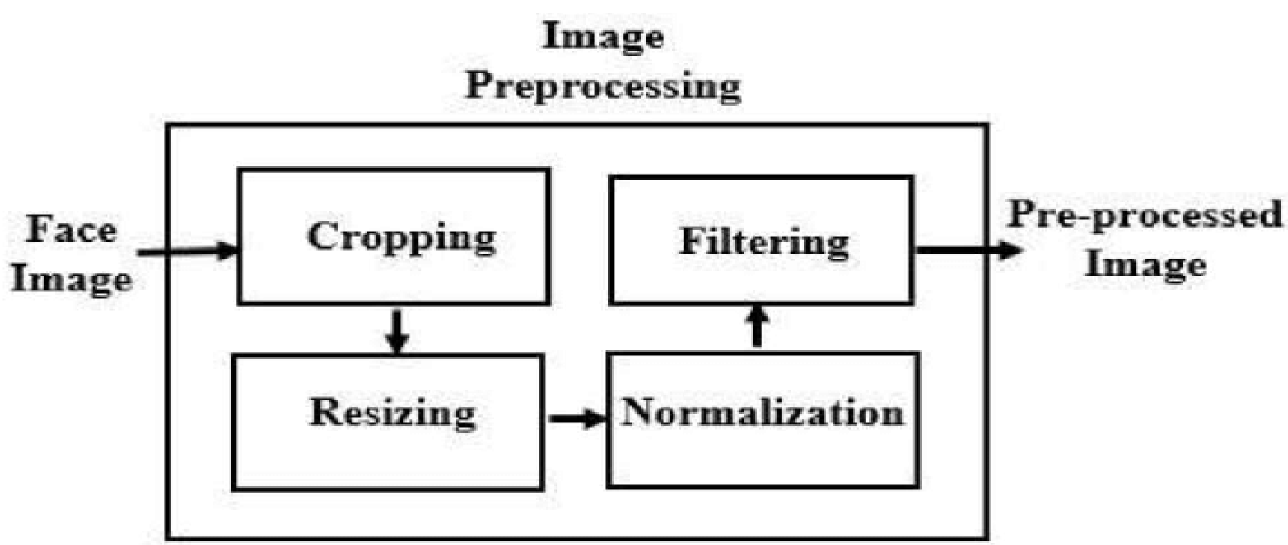

Figure 3.1 Image Preprocessing Steps

Cropping: Focuses on the region of interest by removing unnecessary background, ensuring that only the relevant part of the image is processed further.

Filtering: Enhances the quality of the image by reducing noise, sharpening edges, or applying other techniques to highlight important features.

Resizing: Adjusts the image to a standard size, which is important for ensuring uniformity across all images being analyzed, making subsequent processing steps more consistent.

Normalization: Adjusts the pixel values to a standard range, improving consistency by mitigating variations in lighting and other conditions.

Image transforms

All the image processing approaches discussed thus far operate directly on the pixels of an input image. That is, they work directly in the spatial domain. In some cases, image processing tasks are best formulated by transforming the input images, carrying the specified task in a transform domain, and applying the inverse transform to return to the spatial domain. [2]

Spatial filtering

Used in a broad spectrum of image-processing applications, a solid understanding of filtering principles is important.

Spatial filtering modifies an image by replacing the value of each pixel with a function of the values of the pixel and its neighbors.

If the operation performed on the image pixels is linear, then the filter is called a linear spatial filter. [2]

Otherwise, the filter is a nonlinear spatial filter.

Aliasing

- **Sampling issue:** Occurs when capturing images with insufficient resolution (sampling rate). High-frequency details are lost, leading to artifacts like jagged edges.

Filtering

- **Frequency manipulation:** A technique that alters the detail level (frequencies) in an image. Filters are applied in the frequency domain to selectively remove noise (high frequencies) or enhance features (adjusting specific frequencies).

Convolution

Two functions involve flipping (rotating by 180°) one function about its origin and sliding it past the other.

At each displacement in the sliding process, we perform a computation which, for discrete variables, is a sum of products in the present discussion. [2]

Periodic noise

Can be analyzed and filtered quite effectively using frequency domain techniques. The basic idea is that periodic noise appears as concentrated bursts of energy in the Fourier transform, at locations corresponding to the frequencies of the periodic interference.

The approach is to use a selective filter to isolate the noise. [2]

Segmentation

Segmentation is a process that partitions an image into regions.[2]

Threshold-based Segmentation

- Definition: Divides an image into foreground and background by comparing pixel values to a threshold.
- Use Cases: Simple applications like document scanning.

Edge-based Segmentation

- Definition: Identifies edges where there are significant changes in intensity using edge detection algorithms.
- Use Cases: Detecting objects with well-defined boundaries, such as in object detection.

Region-based Segmentation

- Definition: Groups pixels into regions based on similarity in color, intensity, or texture.
- Use Cases: Segmenting regions with homogeneous properties, useful in medical imaging and satellite imagery.

3.3 What is the difference between thermal and light intensity

Differences between Thermal and Light Intensity:

Nature of the Data:

- **Light Intensity**: Captures the amount of visible light (from the visible spectrum) that is reflected or emitted from objects. It is typically captured using standard cameras (RGB or grayscale).
- **Thermal Intensity**: Captures infrared radiation emitted by objects, which correlates with their temperature. It is captured using thermal cameras.

Information Conveyed:

- **Light Intensity**: Indicates the brightness and color of objects, which can be influenced by lighting conditions and material properties.
- **Thermal Intensity**: Indicates the temperature distribution of objects, providing insights into heat patterns and thermal anomalies.

Similarities in Digital Image Processing:

Feature Extraction:

- Common features like edges, corners, textures, and shapes can be extracted from both light intensity and thermal images.
- Algorithms like the Canny edge detector, HOG (Histogram of Oriented Gradients), and SIFT (Scale-Invariant Feature Transform) can be used on both types of images.

Classification

Classification is a supervised learning concept that basically categorizes a set of data into classes, it is a predictive modeling problem where a class label is predicted for a given example of input data, An Example of classification is trying to learn categorical class such as “red” or “blue” or “yellow”. [4]

Classification models

➢ Random Forest.

➢ Decision Trees.

➢ Logistic Regression.

➢ Support Vector Machines.

If the task you’re doing does not have labeled data, so you need to know that you have two options depending on the task:

3.4 Deep Learning

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers.

These neural networks attempt to simulate the behavior of the human brain albeit far from matching its ability allowing it to "learn" from large amounts of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy.

Deep learning drives many artificial intelligence "AI" applications and services that improve automation, performing analytical and physical tasks without human intervention. Deep learning technology lies behind everyday products and services "such as digital assistants, voice-enabled TV remotes, and credit card fraud detection" as well as emerging technologies "such as self-driving cars".

Deep learning attempts to mimic the human brain—albeit far from matching its ability enabling systems to cluster data and make predictions with incredible accuracy. [5]

Types of deep learning models

- **CNNs (Convolutional Neural Networks):** Extract features from images through interconnected processing layers, excelling in object recognition and classification.
- **GANs (Generative Adversarial Networks):** Employ competing networks to create new, realistic images (inpainting, data augmentation).
- **RNNs (Recurrent Neural Networks):** Analyze sequential image data like video frames for tasks like action recognition or anomaly detection.

- **Transformers:** Powerful for processing large amounts of image data, used for image classification and object detection.

3.4.1 Why Using Deep Learning Model

Deep learning models are used for a variety of reasons, primarily because they excel at tasks that involve processing and understanding complex data. Here are some key reasons why deep learning models are popular:

- **Feature Learning**: Deep learning models can automatically learn to extract relevant features from raw data, reducing the need for manual feature engineering.
- **Highly Scalable**: They can scale with data size and complexity, often performing better with more data, which is crucial in today's era of big data.
- **Flexibility**: Deep learning models can be applied across various domains, from image and speech recognition to natural language processing and autonomous systems.
- **State-of-the-Art Performance**: They often achieve state-of-the-art performance in many tasks, surpassing traditional machine learning algorithms.
- **Continuous Improvement**: With ongoing research and advancements, deep learning models continue to improve in accuracy and efficiency.
- **Adaptability**: They can adapt to different types of data and tasks through architectures like convolutional neural networks (CNNs) for images, recurrent neural networks (RNNs) for sequential data, and transformers for natural language processing.
- **Automation**: Deep learning models can automate complex decision-making processes that previously required human intervention, leading to efficiency gains in various applications.

Deep learning layers

- **Input Layer**: Receives data for processing.

- **Hidden Layers**: Perform computations and feature extraction (Includes Dense, Convolutional, Recurrent, and Pooling layers).
- **Output Layer**: Produces final predictions or outputs.
- **Activation Functions**: Introduce non-linearity (e.g., ReLU, sigmoid).
- **Normalization Layers**: Normalize and stabilize input or hidden data.

These layers work together to process and learn from input data, making deep learning models effective for a wide range of tasks like image recognition, natural language processing, and more.

As shown in Figure 3.2, which illustrates the deep neural network layers.

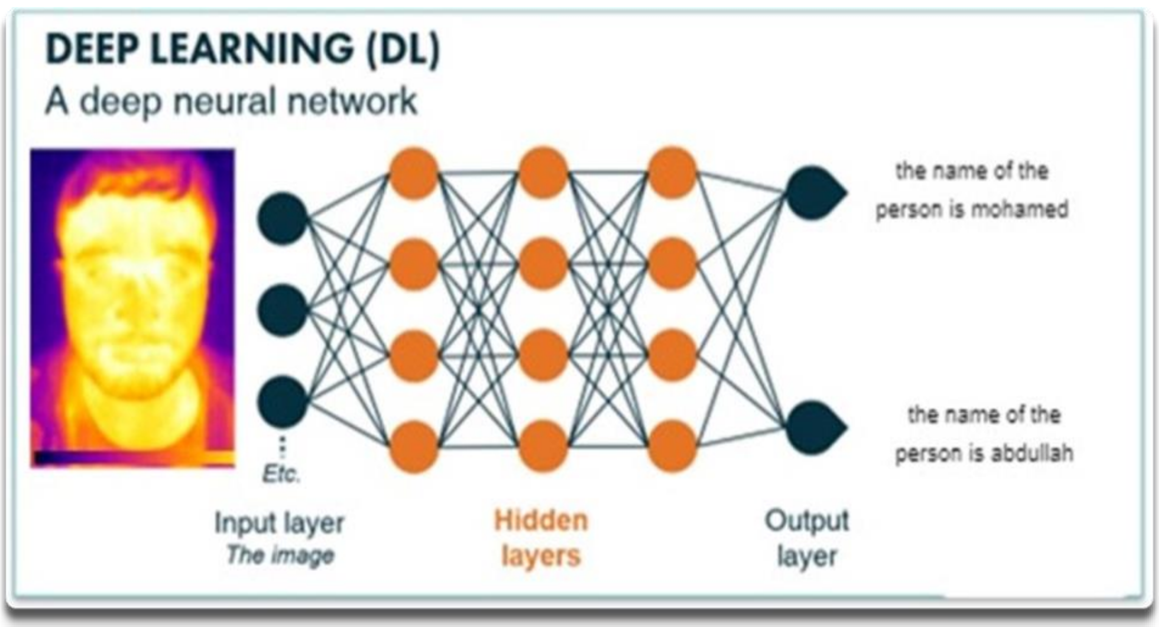

Figure 3.2 Deep neural network layers

3.5 Activation Functions

- In artificial neural network, each neuron computes a weighted sum of its inputs.
- This sum passes through an activation function, determining if the neuron activates.
- The activation function decides the relevance of neuron inputs in predictions.
- It transforms the summed input into an output for subsequent layers or as final output.

3.5.1 Can we do without an activation function?

We understand that using an activation function introduces an additional step at each layer during the forward propagation.

Now the question is if the activation function increases the complexity so much, imagine a neural network without the activation functions. In that case, every neuron will only be performing a linear transformation on the inputs using the weights and biases.

Although linear transformations make the neural network simpler, this network would be less powerful and will not be able to learn the complex patterns from the data.

"A neural network without an activation function is essentially just a linear regression model." [4]

Thus, we use a nonlinear transformation to the inputs of the neuron and this non-linearity in the network is introduced by an activation function.

3.5.2 Activation functions can be divided into three types

- Linear Activation Function
- Binary Step Function
- Non-linear Activation Functions

Linear Activation Function

The linear activation function, often called the identity activation function, is proportional to the input. The range of the linear activation function will be

(-∞, ∞).

The linear activation function simply adds up the weighted total of the inputs and returns the result.

Linear Activation Function

$$F(x)=x$$

Equation 3. 1, Linear Activation Function

Binary Step Activation Function

A threshold value determines whether a neuron should be activated or not activated in a binary step activation function.

The activation function compares the input value to a threshold value.

If the input value is greater than the threshold value, the neuron is activated.

It’s disabled if the input value is less than the threshold value, which means its output isn’t sent on to the next or hidden layer.

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

Equation 3. 2, binary step activation function

Non-linear Activation Functions

The Non-linear activation functions are the most-used activation functions.

They make it uncomplicated for an artificial neural network model to adapt to a variety of data and to differentiate between the outputs.

Non-linear activation functions allow the stacking of multiple layers of neurons, as the output would now be a non-linear combination of input passed through multiple layers.

➢ **Sigmoid**

Sigmoid accepts a number as input and returns a number between 0 and 1.

It's simple to use and has all the desirable qualities of activation functions: nonlinearity, continuous differentiation, monotonicity, and a set output range.

This is mainly used in binary classification problems.

This sigmoid function gives the probability of the existence of a particular class.

$$f(x) = \frac{1}{1 + e^{x}}$$

Equation 3. 3, sigmoid

➢ **Hyperbolic Tangent (TanH)**

TanH compresses a real-valued number to the range [-1, 1]. It's non-linear, but it's different from Sigmoid, and its output is **zero-centered**.

The main advantage of this is that the negative inputs will be mapped strongly to the negative and zero inputs will be mapped to almost zero in the graph of TanH.

$$f(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}$$

Equation 3. 4, (TanH)

➢ **Rectified Linear Unit (ReLU)**

ReLU stands for Rectified Linear Unit and is one of the most commonly used activation function in the applications.

It's solved the problem of vanishing gradient because the maximum value of the gradient of ReLU function is one.

It also solved the problem of saturating neuron, since the slope is never zero for ReLU function.

The range of ReLU is between **0 and infinity.**

$$\mathbf{f(x) = max(0, x)}$$

Equation 3. 5, (ReLU)

➢ **Leaky ReLU**

Is an upgraded version of the ReLU activation function to solve the dying ReLU problem, as it has a small positive slope in the negative area.

But the consistency of the benefit across tasks is presently ambiguous.

$$f(x) = \max(0.1x, x)$$

Equation 3. 6, (Leaky ReLu)

➢**SoftMax**

A combination of many sigmoid is referred to as the SoftMax function.

It determines relative probability. Similar to the sigmoid activation function, the SoftMax function returns the probability of each class/label.

In multi-class classification, SoftMax activation function is most commonly used for the last layer of the neural network.

The SoftMax function gives the probability of the current class with respect to others.

This means that it also considers the possibility of other classes too.

$$softmax(xi) = \frac{\exp(xi)}{\sum_j \exp(xj)}$$

Equation 3. 7, softmax

3.6 Convolutional Neural Network “CNN”

So images has so many data and details on it thus, machine learning couldn’t work with it, also a neural network couldn’t handle the pixel details and learn from it in an accurate way, that’s why there was a convolutional neural network which is an extend of neural networks that was especially designed to work with images and identifying objects.

“A CNN is a kind of network architecture for deep learning algorithms and is specifically used for image recognition and tasks that involve the processing of pixel data”. [6]

As shown in Figure 3.3, which illustrates the Convolutional neural network:

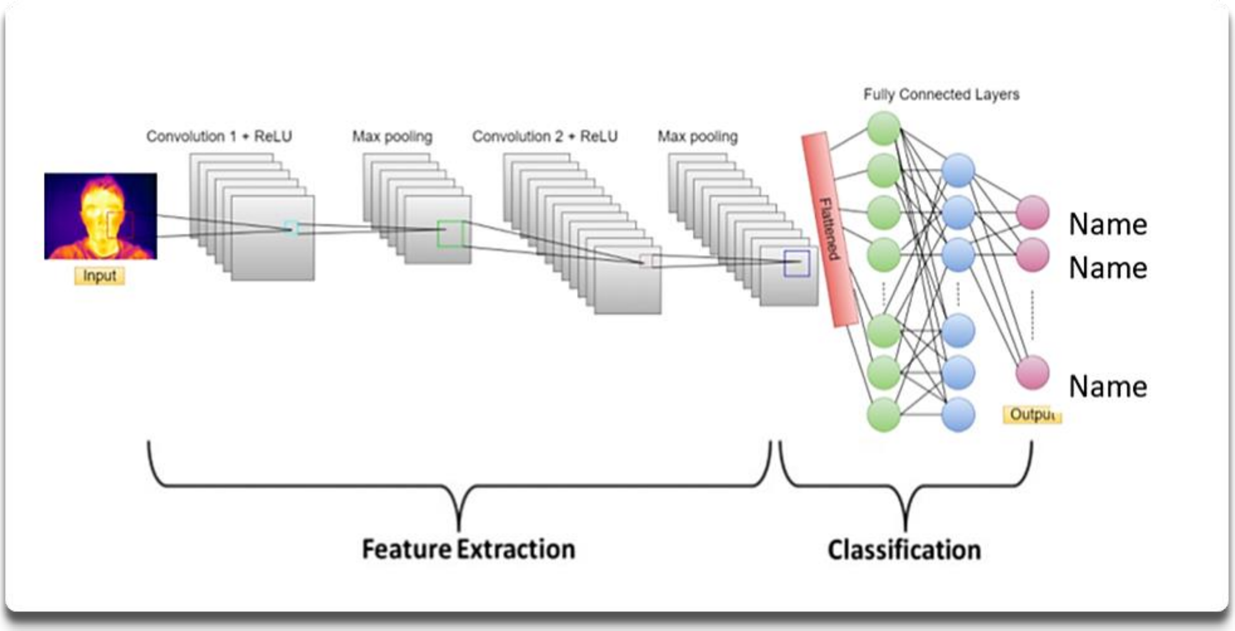

Figure 3.3 Convolutional neural network

3.6.1 Applications of CNN

CNNs are used in a wide variety of applications, including:

- Image classification.
- Object detection.
- Image segmentation.
- Facial recognition.
- Medical image analysis.

3.6.2 Why use CNN not CV?

Convolutional Neural Networks “CNNs” are a powerful tool for visual analysis and have become the go-to method for many computer vision tasks.

This is due to several advantages CNNs have over traditional computer vision “CV” techniques.

- **Feature Extraction:**

  CNNs are specifically designed to extract relevant features from images, which is crucial for tasks like image classification, object detection, and image segmentation.

  Traditional CV techniques often rely on hand-crafted features, which can be time-consuming to design and may not be as effective as features learned by CNNs.

- **End-to-End Learning:**

NNs can be trained in an end-to-end manner, meaning they learn to extract features and perform classification or detection directly from the raw image data. This eliminates the need for hand-crafted features and simplifies the training process.

- **Robustness:**

CNNs are relatively robust to variations in lighting, pose, and background, making them more adaptable to real-world applications. They can generalize well to unseen data, leading to better performance in real-world scenarios.

- **Scalability:**

CNNs can be efficiently scaled to process large datasets and high-resolution images, making them suitable for large-scale applications. They can be parallelized on GPUs and other hardware accelerators for faster training and inference.

Due to these advantages, CNNs have become the dominant approach for many computer vision tasks. They have achieved state-of-the-art performance in image classification, object detection, and image segmentation, and are being actively applied in various fields, including healthcare, robotics, autonomous vehicles, and security.

While traditional CV techniques still have their place in certain applications, CNNs have revolutionized computer vision and continue to push the boundaries of what can be achieved with visual analysis.

➢ **CNN Layers**

A deep learning CNN consists of three layers: a convolutional layer, a pooling layer and a fully connected “FC” layer.

The convolutional layer is the first layer while the FC layer is the last. From the convolutional layer to the FC layer, the complexity of the CNN increases.

It is this increasing complexity that allows CNN to successively identify larger portions and more complex features of an image until it finally identifies the object in its entirety. [6]

➢ **Convolutional Layer**

The majority of computations happen in the convolutional layer, which is the core building block of CNN.

As shown in Figure 3.4, which illustrates the convolutional layer.

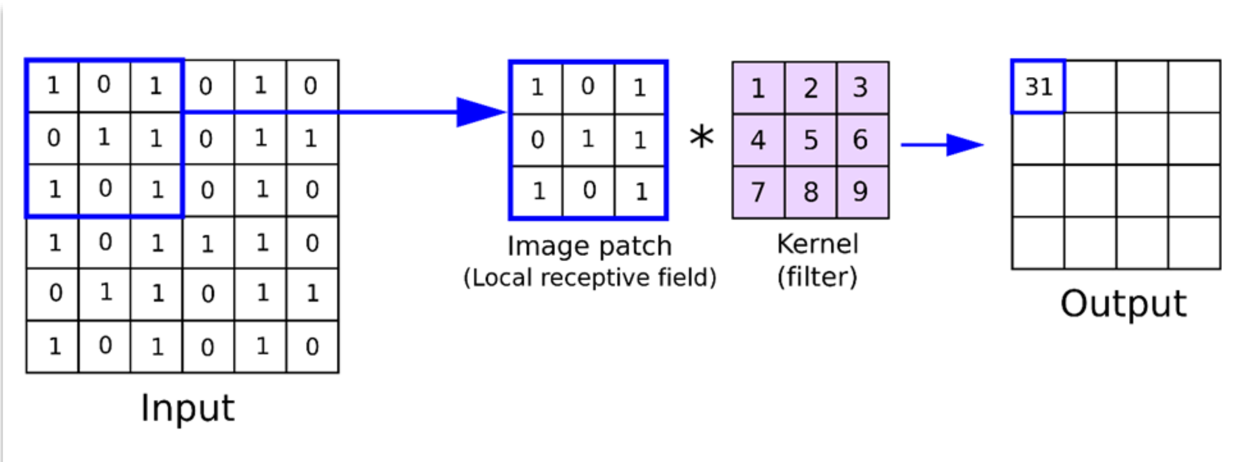

Figure 3.4 Convolutional layer

A second convolutional layer can follow the initial convolutional layer. The process of convolution involves a kernel or filter inside this layer moving across the receptive fields of the image, checking if a feature is present in the image.

Over multiple iterations, the kernel sweeps over the entire image.

After each iteration a dot product is calculated between the input pixels and the filter.

The final output from the series of dots is known as a feature map or convolved feature.

Ultimately, the image is converted into numerical values in this layer, which allows CNN to interpret the image and extract relevant patterns from it.

➢ Pooling Layer

pooling layers are used to progressively reduce the spatial dimensions (width and height) of the input volume for each convolutional layer, thereby reducing the number of parameters and computation in the network. Pooling helps in controlling overfitting and can make the model more robust to variations in the input data.

As shown in Figure 3.5, which illustrates the pooling layer.

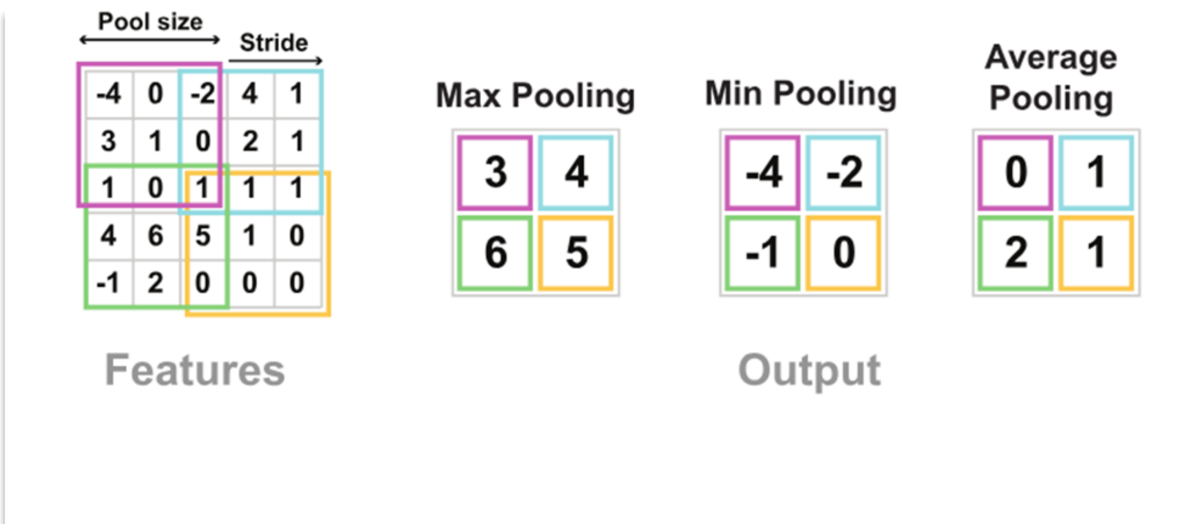

Figure 3.5 pooling layer

There are several types of pooling layers commonly used:

1. **Max Pooling**:
    - Max pooling takes the maximum value from each patch of the feature map defined by the pooling size and stride.

- **How it works**: For each pooling region, the maximum value is retained while discarding the rest. This helps in preserving the most important features and reducing spatial dimensions.
- Max pooling is effective in capturing the most activated features and is robust to small variations in the input.

2. **Average Pooling**:
   - Average pooling computes the average value from each patch of the feature map.
   - **How it works**: It calculates the average intensity of the pixels in the pooling window. This can help in creating a more generalized representation of the input.
   - Average pooling is simpler than max pooling but may lose some subtle features preserved by max pooling.
3. **Min Pooling**:
   - Min pooling takes the minimum value from each patch of the feature map.
   - **How it works**: It selects the minimum value within each pooling window. This can be useful in specific cases where detecting the minimum activation is meaningful.
   - Min pooling is less common compared to max and average pooling but can be used in scenarios where finding the smallest activation is relevant.

➢ **Fully Connected Layer**

The FC layer is where image classification happens in the CNN based on the features extracted in the previous layers.

As shown in Figure 3.6, which illustrates the fully connected layer.

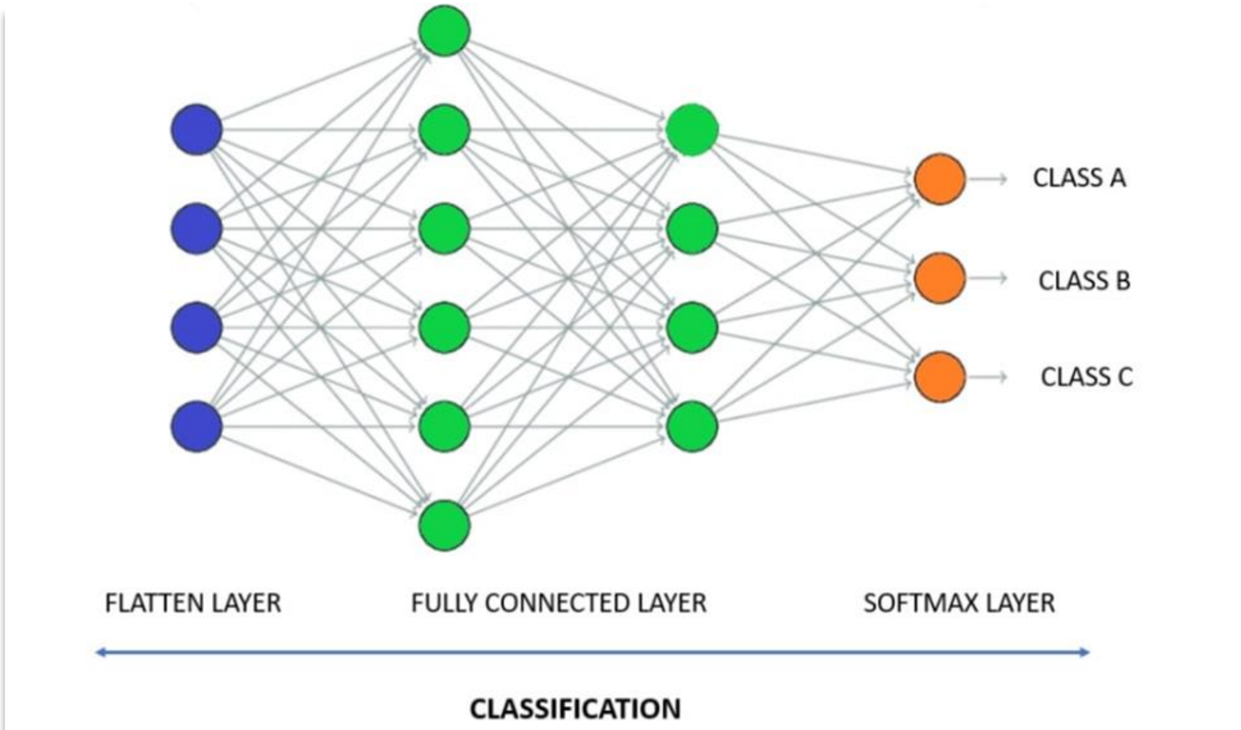

Figure 3.6 fully connected layer

Here, fully connected means that all the inputs or nodes from one layer are connected to every activation unit or node of the next layer.[6]

All the layers in CNN are not fully connected because it would result in an unnecessarily dense network.

It also would increase losses and affect the output quality, and it would be computationally expensive.

The difference between forward propagation and backward propagation

- **Forward propagation** in a neural network involves transmitting input data through the network's layers. Each layer calculates a weighted sum of inputs, adds a bias, and applies an activation function to produce its output. This process continues sequentially until the final layer generates the network's prediction.
- **Back propagation**, on the other hand, begins with computing the error between the network's prediction and the actual target using a loss function. It then calculates the gradient of this error with respect to each parameter (weights and biases) using the chain rule of calculus. These gradients are propagated backward through the network, allowing each layer to adjust its parameters proportionally to their contribution to the error. This iterative adjustment process optimizes the network's parameters to minimize prediction errors over time, enabling it to learn and improve its performance.

As shown in Figure 3.7, which illustrates the forward propagation and backward propagation:

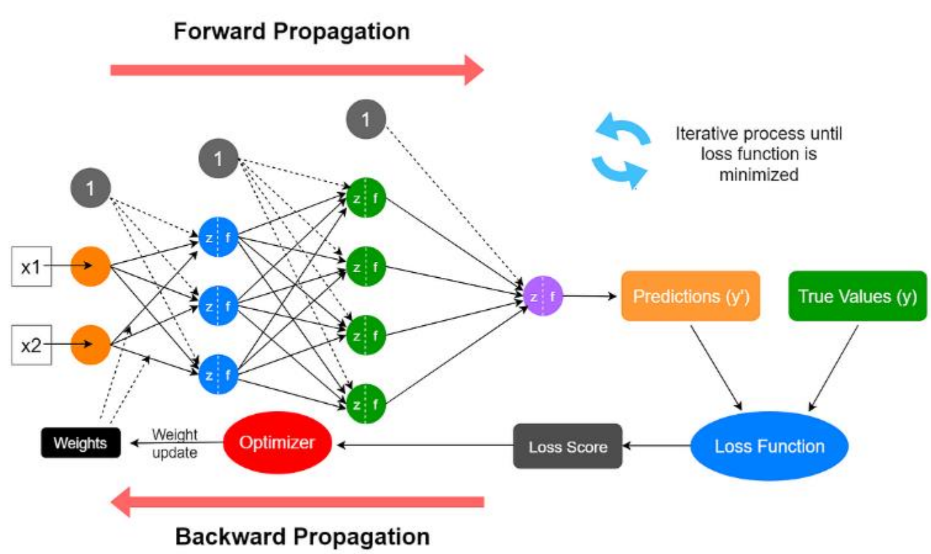

Figure 3.7 forward propagation and backward propagation

Epochs and Batch Size in CNNs

Epochs:

An epoch in the context of Convolutional Neural Networks (CNNs) refers to one complete pass through the entire training dataset. During each epoch, the model processes every training sample exactly once, updating its weights based on the error in its predictions. Typically, training a CNN involves multiple epochs to allow the model to learn and improve its performance incrementally.

Why multiple epochs?

Training for multiple epochs allows the model to refine its weights gradually. Early epochs might show significant improvements, while later epochs contribute to fine-tuning the model's performance.

Batch Size:

Batch size is the number of training samples processed before the model's internal parameters (weights) are updated. Instead of updating weights after each individual sample (which can be computationally expensive), samples are grouped into batches.

Why use batches?

Using batches makes the training process more efficient by leveraging vectorized operations and helps smooth out noisy gradient updates, leading to more stable convergence.

Optimizers in CNNs

Optimizers are algorithms or methods used to change the attributes of the neural network such as weights and learning rate in order to reduce the losses. Optimization algorithms are at the heart of neural network training and choosing the right one can make a significant difference in training performance and convergence speed.

How Optimizers Work:

Optimizers adjust the model's weights in response to the computed gradients of the loss function with respect to the weights. The main goal is to minimize the loss function. Here are some key concepts related to optimizers:

Gradient Descent: The foundational algorithm for most optimizers. It involves calculating the gradient of the loss function and updating the weights in the opposite direction of the gradient.

- **Learning Rate**: A hyperparameter that determines the step size during the weight update. A higher learning rate might speed up training but risk overshooting minima, while a lower learning rate might make training slow but more stable.

Types of Optimizers:

Stochastic Gradient Descent (SGD):

- Updates weights using a single sample (or a small batch) at a time.
- Simple and effective but can be noisy and may struggle with complex landscapes.

Momentum:

- Builds on SGD by adding a fraction of the previous update to the current update.
- Helps accelerate gradients vectors in the right directions, leading to faster converging.

Nesterov Accelerated Gradient (NAG)

- A variant of momentum that looks ahead to where the momentum is leading and makes corrections.

Adagrad:

- Adjusts the learning rate for each parameter individually, performing smaller updates for parameters associated with frequently occurring features.
- Good for sparse data but can lead to overly small learning rates.

Adadelta:

- An extension of Adagrad that seeks to reduce Adagrad's aggressive, monotonically decreasing learning rate.
- Accumulates past gradients but uses a window instead of all past gradients.

RMSprop:

- An unpublished, adaptive learning rate method proposed by Geoff Hinton.
- Works similarly to Adadelta by normalizing gradients using a moving average of the squared gradients.

Adam (Adaptive Moment Estimation):

- Combines the advantages of two other extensions of stochastic gradient descent, AdaGrad and RMSProp.
- Maintains separate learning rates for each parameter, which are adjusted based on the first and second moments of the gradients.

AdaMax:

- A variant of Adam based on the infinity norm.

Nadam:

- Combines Adam with Nesterov momentum.

As shown in Figure 3.8, which illustrates the *Comparison of Adam to Other Optimizers*

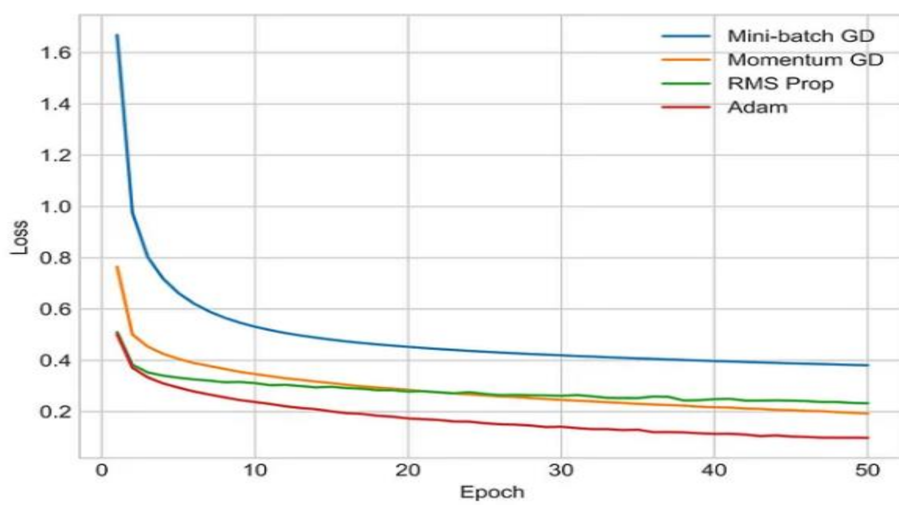

Figure 3.8 *Comparison of Adam to Other Optimizers*

1D vs 2D Convolutional Neural Networks (CNNs):

1D CNNs:

- **Input:** Processes 1-dimensional data (e.g., time series, text).
- **Operation:** Uses 1D kernels sliding over the input sequence.
- **Use Cases:** Suitable for tasks like NLP, speech recognition.

2D CNNs:

- **Input:** Processes 2-dimensional data (e.g., images).
- **Operation:** Uses 2D kernels sliding over the input image.
- **Use Cases:** Ideal for tasks in computer vision like image classification, object detection.

Key Differences:

- **Data Dimension:** 1D CNNs handle sequential data(text), while 2D CNNs handle spatial data (images).
- **Kernel Shape:** 1D CNNs use 1D kernels, whereas 2D CNNs use 2D kernels.

As shown in Figure 3.9, which illustrates 1D vs 2D (CNN)

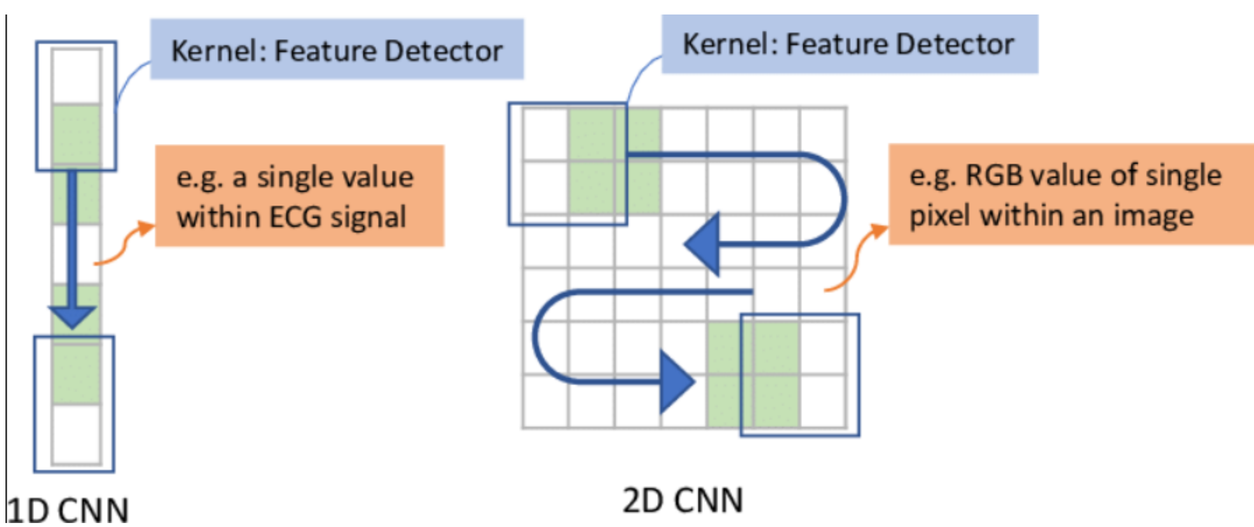

Figure 3.9 *1D vs 2D (CNN)*

3.6.3 Overfitting

Data overfitting occurs when a model performs well on training data but fails to generalize to new data.

It happens when the model captures noise or random fluctuations instead of the underlying pattern.

Overfitting can lead to poor performance on unseen or future data.

3.6.4 Regularization

Regularization refers to techniques that are used to calibrate machine learning models in order to minimize the adjusted loss function and prevent overfitting.

➢ **Types of regularization**

- Dropout Method.
- Early Stopping Method.
- Cross Validation Method.
- L1 and L2 Regularization method.
- Data Augmentation.

3.6.4.1 Dropout Method

Is a regularization technique that randomly drops out a proportion of neural network units during training, preventing over-reliance on specific features and improving generalization.

As shown in Figure 3.10, which illustrates dropout*:*

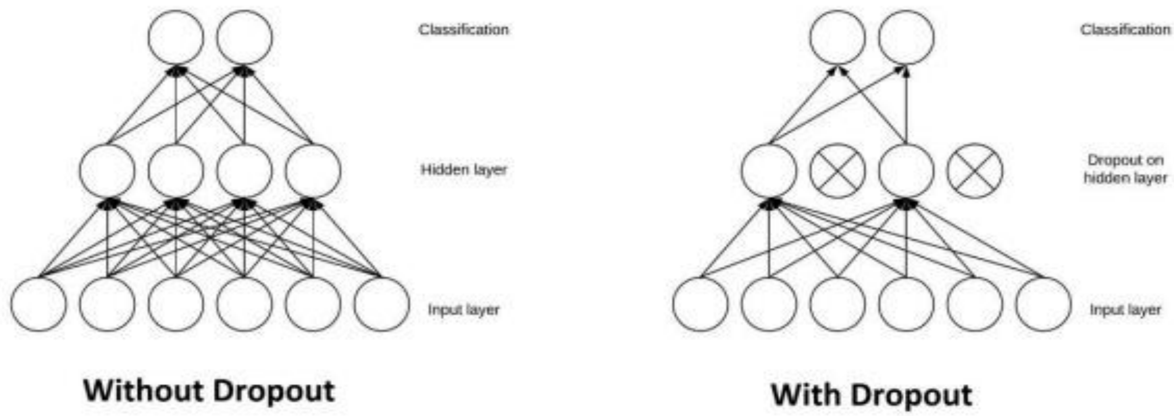

Figure 3.10 *dropout*

3.6.4.2 Early stopping

Is a technique where training is stopped when the performance on a validation set starts to degrade, preventing the model from overfitting by capturing the best accuracy for the model.

As shown in Figure 3.9, which illustrates *Early stopping:*

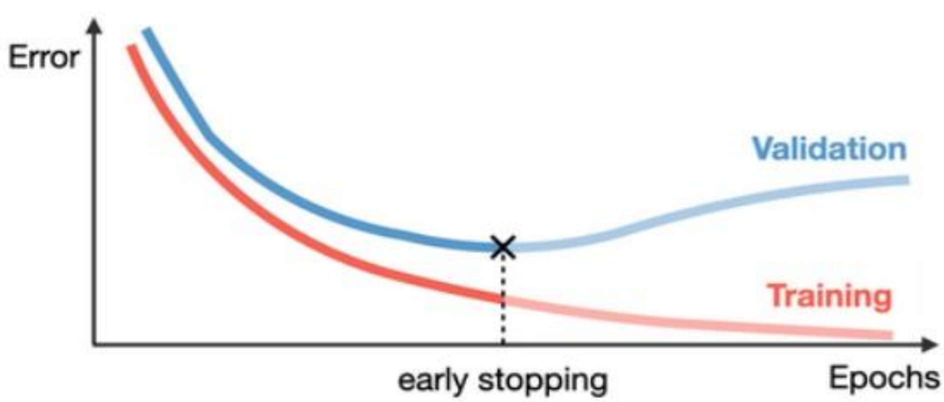

Figure 3.11 *Early stopping*

3.6.4.3 Cross validation

Is a technique used in machine learning to evaluate the performance of a model on an independent dataset. It helps to prevent overfitting and provides insights into the model's generalization ability.

- **Leave-One-Out Cross Validation**
  - Uses n-1 samples from the original dataset as the training set and the remaining sample as the validation set.
  - Repeats this for every sample in the dataset.
- **K-Fold Cross Validation**
  - Divides the data into k subsets “folds” and trains the model k times, each time using a different fold as the validation set.

3.6.4.4 L1 regularization

Also known as Lasso regularization

$$\text{Modified loss function} = \text{Loss function} + \lambda \sum_{i=1}^{n} |W_i|$$

Equation 3. 8, L1 regularization

When to use it:

- You want to select a small number of important features.
- You have a lot of irrelevant or noisy features.
- You want a model that is robust to outliers.

3.6.4.5 L2 regularization

Also known as Ridge regularization

$$\text{Modified loss function} = \text{Loss function} + \lambda \sum_{i=1}^{n} W_i^{\,2}$$

Equation 3. 9, L2 regularization

When to use it:

- You want a stable and reliable model.
- You want to avoid overfitting caused by complex models.
- You have a lot of relevant features.

3.7 Recurrent Neural Network “RNN”

A Recurrent Neural Network (RNN) is a type of artificial neural network designed to process sequential data by using information from previous inputs to influence the current output. This makes RNNs particularly useful for tasks like language translation, speech recognition, and time series prediction, where the order of data points is important. Unlike traditional neural networks, RNNs have a "memory" that captures information from earlier inputs in the sequence.

As shown in Figure 3.12, which illustrates the Recurrent Neural Network

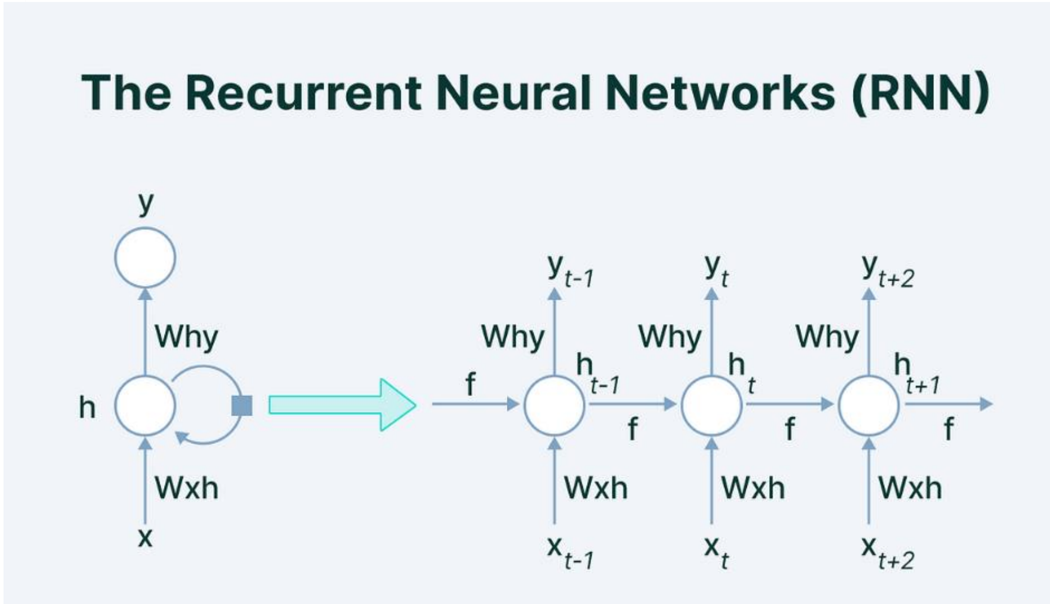

Figure 3.12 *(RNN)*

- Designed to handle sequential data or time series data, ideal for tasks where data points are dependent on previous ones.

Common Uses:

- Ordinal or temporal problems.
- Applications: language translation, natural language processing (NLP), speech recognition, image captioning.

Incorporation into Popular Applications:

- Siri, voice search, Google Translate.

Learning Mechanism:

- Uses training data to learn patterns, similar to other neural networks.
- Training involves techniques like gradient descent and backpropagation through time (BPTT).

Memory Feature:

- Captures information from previous inputs to influence current outputs.
- Uses hidden states to store this information, updating them at each time step.

Sequential Dependency:

- Outputs depend on prior elements within the sequence, unlike traditional neural networks that treat inputs and outputs as independent.

Limitation of Unidirectional RNNs:

- Cannot account for future events in their predictions.
- Bidirectional RNNs process data in both forward and backward directions, allowing the network to consider future context for more accurate predictions.

3.8 The Steps That Lead to Successful CNN Model

Face recognition using thermal image the following:

Performing face recognition using thermal imaging involves several steps. Here's a high-level overview of the process:

Obtain Thermal Images:

Capture thermal images using a thermal camera or a device that can capture thermal data, such as an infrared camera.

Face Detection:

Use a face detection algorithm or library to locate faces in the thermal images. There are various face detection algorithms available, such as Haar cascades, HOG + SVM, or deep learning-based approaches like MTCNN or YOLO.

Face Alignment:

Align the detected faces to a standardized pose or position. This step helps to normalize the face images for further analysis. Techniques like facial landmark detection or affine transformations can be used for face alignment.

Feature Extraction:

Extract meaningful features from the aligned face regions. Commonly used techniques for face feature extraction include Local Binary Patterns (LBP), Histogram of Oriented Gradients (HOG), or deep learning-based methods such as Convolutional Neural Networks (CNN).

Face Recognition:

Train a face recognition model using the extracted features. This step typically involves training a machine learning model (e.g., Support Vector Machine, k-Nearest Neighbors, or a deep learning model) to recognize individuals based on their face features.

Classification:

Once the face recognition model is trained, use it to classify new faces based on their features. The model will associate each face with the identity it has been trained on.

It is worth noting that thermal imaging for face recognition may have some limitations compared to visible light-based face recognition.

Thermal images capture heat patterns rather than visual appearance, which can be influenced by factors like lighting conditions, variations in thermal signatures, and occlusions.

Additionally, the availability of thermal face datasets may be more limited compared to visible light datasets. Specific implementations can vary depending on the chosen algorithms and libraries.

Chapter 4
System Implementation

Chapter Four System Implementation

4.1 Proposed System Solution

The proposed system for thermal face recognition involves several key steps, each crucial for developing an efficient and accurate recognition system. Below is a detailed outline of the steps involved:

4.1.2 Dataset Collection

Thermal Image Dataset Creation: We began by creating a comprehensive dataset of thermal face images. This dataset was curated to include a diverse range of subjects, The dataset aims to provide a robust foundation for training and testing the recognition model.

4.1.3 Data Augmentation

Enhancing dataset diversity: To increase the robustness and generalizability of the model, data augmentation techniques were applied to the thermal images. This step helps in preventing overfitting and improves the model's ability to generalize to new, unseen data.

4.1.4 Data Preprocessing

Preprocessing involved standardizing the thermal images to ensure uniformity across the dataset. This step ensured that all images are of the same dimension and focused on the relevant facial region, which is critical for accurate feature extraction.

4.1.5 Feature Extraction

Extracting Relevant Features from Thermal Images: The next step involved extracting relevant features from the preprocessed thermal images. Various techniques were investigated, with a focus on methods tailored to the unique characteristics of thermal images. The selected feature extraction method plays a crucial role in ensuring the model can accurately distinguish between different faces.

4.1.6 Model Training and Classification

Training Convolutional Neural Network (CNN): A specialized Convolutional Neural Network (CNN) architecture was designed and trained using the preprocessed dataset. The model's parameters were optimized through rigorous training, involving techniques like parameter tuning and adjusting the layers' order to enhance its performance.

4.1.7 System Testing and Evaluation

The final model was rigorously tested in various real-world scenarios, including different lighting conditions and subject positions. This step ensured that the system could perform reliably under diverse conditions.

4.1.8 User Interface Development

Integration with User Interface: To ensure practical usability, a user interface was developed to interact with the face recognition system. The trained model was seamlessly integrated into this interface, enabling efficient and user-friendly face recognition in various environments.

4.2 System Block Diagram

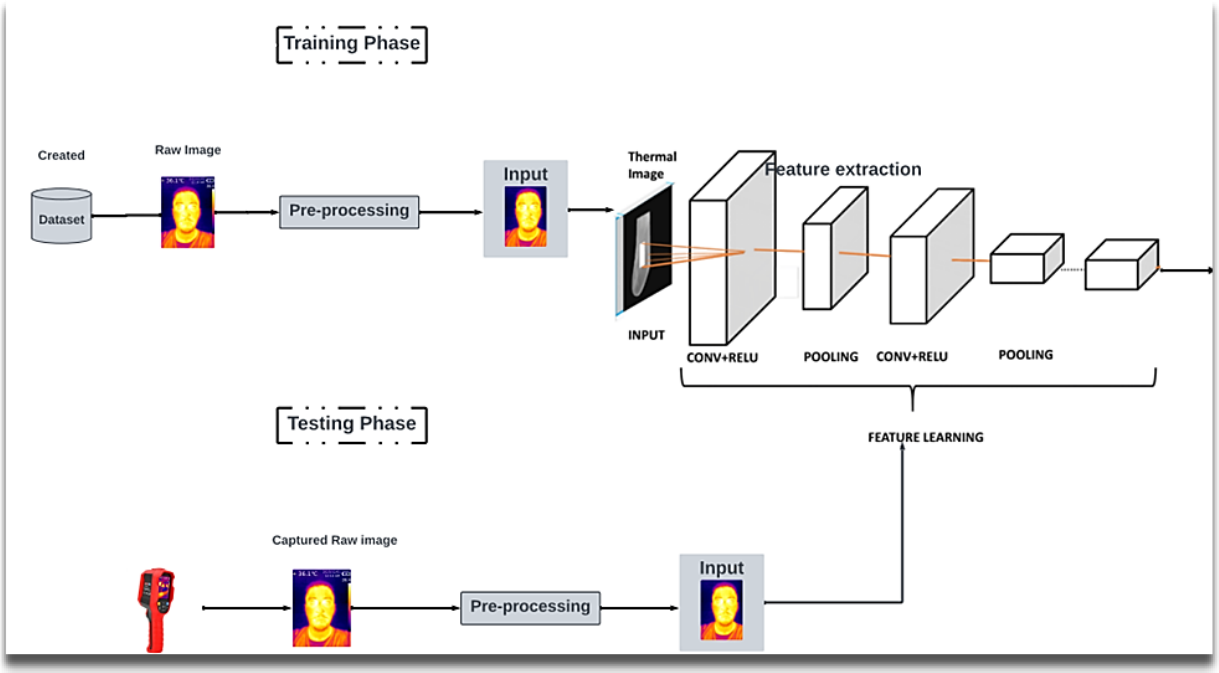

Figure 4.1 System Block Diagram

4.3 Tools & Libraries

Tools:

Jupyter

An interactive computing environment that allows for creating and sharing documents that contain live code, equations, visualizations, and narrative text.

Python (version 3.11.9)

A versatile programming language used for developing and running the thermal face recognition system, known for its readability and robust library support.

Kaggle

A platform providing access to a vast repository of datasets and an environment for running machine learning experiments and sharing code.

Libraries:

Cv2: A library for computer vision tasks, including image processing.

Keras: A high-level API for building and training neural networks, running on top of TensorFlow.

TensorFlow: An open-source platform for machine learning, supporting the development, training, and deployment of machine learning models.

4.4 Dataset

In constructing the dataset, we utilized the UTI-165K, comprising images of 63 distinct individuals captured from 9 different angles.
Each angle captured 3 photographs per individual, resulting in a comprehensive dataset totaling 1,701 images.

To illustrate, the nine angles included:

- Top-Left, Top-Center, Top-Right
- Middle-Left, Middle-Center, Middle-Right
- Bottom-Left, Bottom-Center, Bottom Right

as shown in Figure 4.3, which illustrates the angles of the captured dataset:

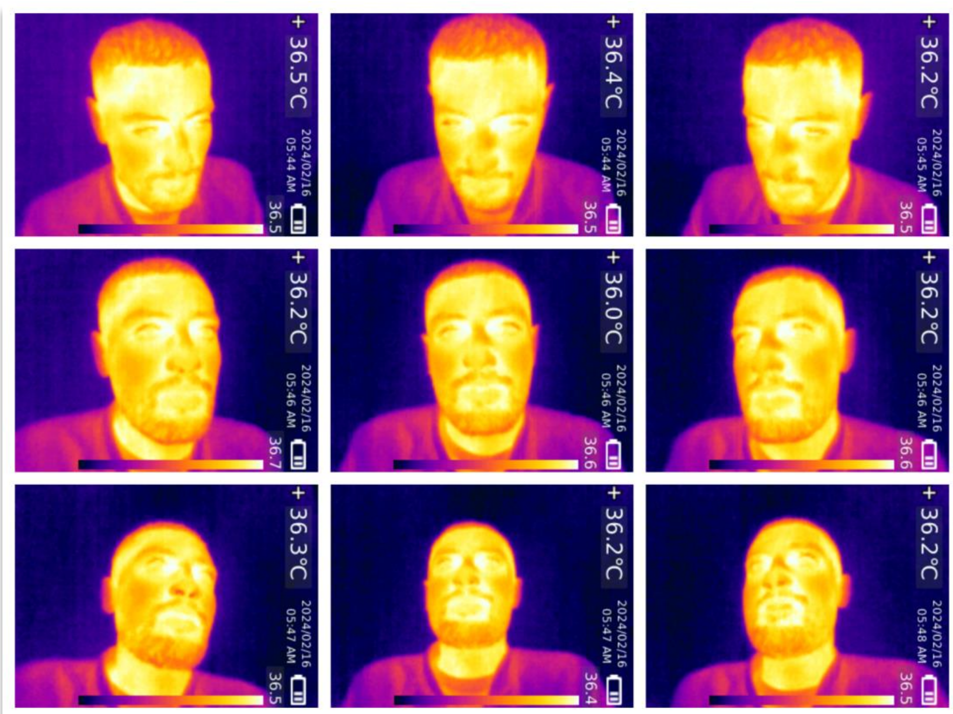

Figure 4.2 The 9 angles of the constructed dataset

Ensuring diverse visual representation of each individual across the dataset, this variety in angles enriches the dataset with different viewpoints, enhancing the robustness and generalizability of the convolutional neural network (CNN) model trained on this data.

4.5 Data Augmentation

4.5.1 What is Data Augmentation

Data augmentation is a technique of artificially increasing the training set by creating modified copies of a dataset using existing data, it includes making minor changes to the dataset or using deep learning to generate new data points.

4.5.2 Data Augmentation Techniques

As shown in Table 4.1, which illustrates some data augmentation techniques:

Table 4.1 Data Augmentation Techniques

Technique	Description	Purpose
Rotation	Rotating the image by a certain angle around a central point.	Creates images from different viewpoints, improving model robustness to variations in object orientation.
Flipping	Mirroring the image horizontally (left-right) or vertically (up-down).	Increases data variety and helps the model learn features independent of object orientation.
Random Cropping	Randomly extracting a sub-region of the image.	Trains the model to recognize objects even when they are not centered or occupy the entire image frame, mimicking real-world scenarios.
Zooming	Scaling the image up (zooming in) or down (zooming out).	Trains the model to recognize objects at different scales and distances.

So, to enhance our limited dataset, we applied several data augmentation techniques. These techniques artificially increased the number of training samples by creating modified versions of the original images, thus improving the robustness and generalizability of our model. We used three main techniques: **rotation**, **flipping** and **random cropping**.

-Rotation

We rotated each photo at intervals from 0 to 320 degrees, generating three photos in total. This technique helps the model become invariant to different orientations of the face.

```
datagen1 = ImageDataGenerator(rotation_range=320)
```

Figure 4.3 Result of rotation technique

-Flipping

We flipped each photo with horizontal flip and saved it. This augmentation ensures that the model can recognize faces regardless of horizontal orientation.

```
datagen2 = ImageDataGenerator(horizontal_flip=True)
```

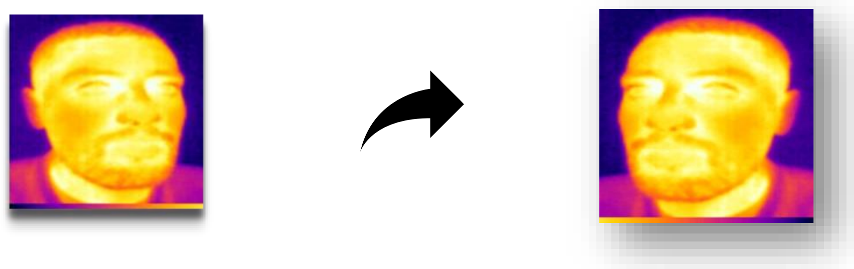

Figure 4.4 Result of flip technique

-Random Cropping

We applied random cropping to the photos, generating three photos in total. This technique involves cropping random portions of the images, which helps the model learn to focus on different parts of the face.

```
x = random.randint(0, width - crop_width)
y = random.randint(0, height - crop_height)
cropped_image = image[y:y + crop_height, x:x + crop_width]
```

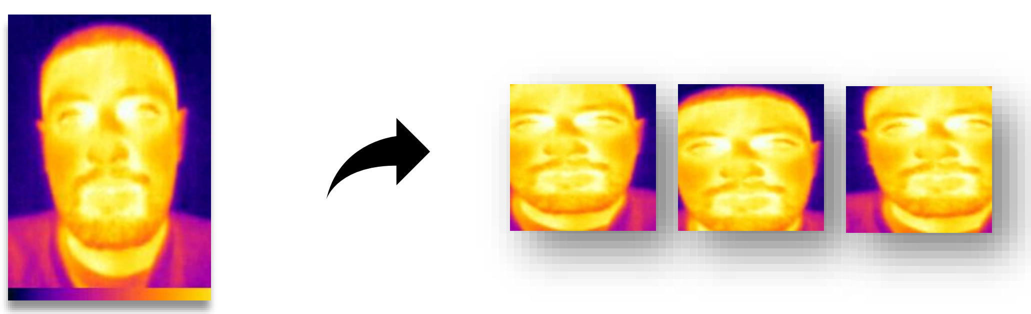

Figure 4.5 Result of random crop technique

By incorporating these data augmentation techniques, we significantly increased the variability in our dataset by increasing it from 1,701 to 13,608 photos, thereby enhancing the model's ability to generalize to new, unseen data.

4.6 Preprocessing the Dataset

4.6.1 Cropping

In the preprocessing stage, we employed cropping to refine the dataset by focusing on the regions of interest, specifically the faces. This technique involved cropping a portion of each photo that included the face of the individual to remove unnecessary background information, thereby enhancing the relevance of the data for the task of face recognition.

4.6.2 Face Detection for Cropping

To automate and accurately perform the cropping, we utilized a pre-trained model, YOLOv5, for face detection. YOLOv5 is known for its efficiency and accuracy in detecting objects within an image. By applying YOLOv5, we obtained the coordinates of the bounding boxes around the faces in each image. Using these coordinates, we cropped the images to isolate the face, ensuring that the cropped images contained only the necessary facial information for the recognition task.

```
cropped=img[y-int(0.5*detection_height):+int(0.5*detection_height),x-
int(0.5*detection_width):x+int(0.5*detection_width)]
```

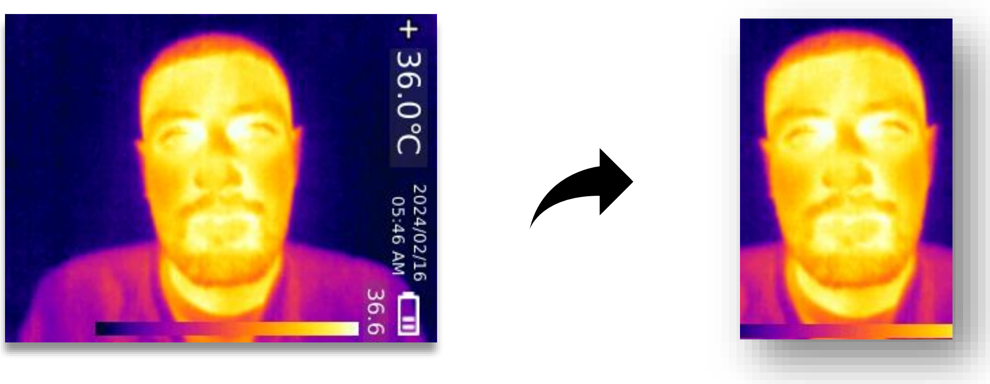

Figure 4.6 Result of crop technique

4.6.3 Resizing

The second preprocessing technique we implemented was resizing. After cropping the images to focus on the faces, we resized each cropped image to a dimension of 150x150 pixels. This size was chosen as it strikes a balance between reducing the computational power required for training the model and maintaining sufficient detail in the images for accurate face recognition. By resizing the images to 150x150, we optimized the dataset to enhance the efficiency of the training process without compromising the quality of the data. This step ensured that the model could process the images more quickly and effectively, facilitating faster and more efficient training cycles.

```
cropped=img[y-int(0.5*detection_height):+int(0.5*detection_height),
x-int(0.5*detection_width):x+int(0.5*detection_width)]
```

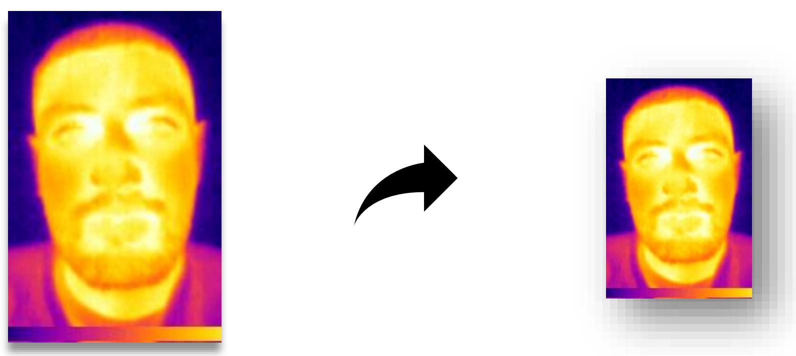

Figure 4.7 Result of resize technique

By incorporating these preprocessing techniques, we ensured that the dataset was optimized for training a robust face recognition model.

4.7 CNN Model Implementation

4.7.1 CNN Model Architecture

As in Figure 4.9, this flowchart shows how the CNN model has been implemented step by step from the Image as an input all the way through the training process to the evaluation process.

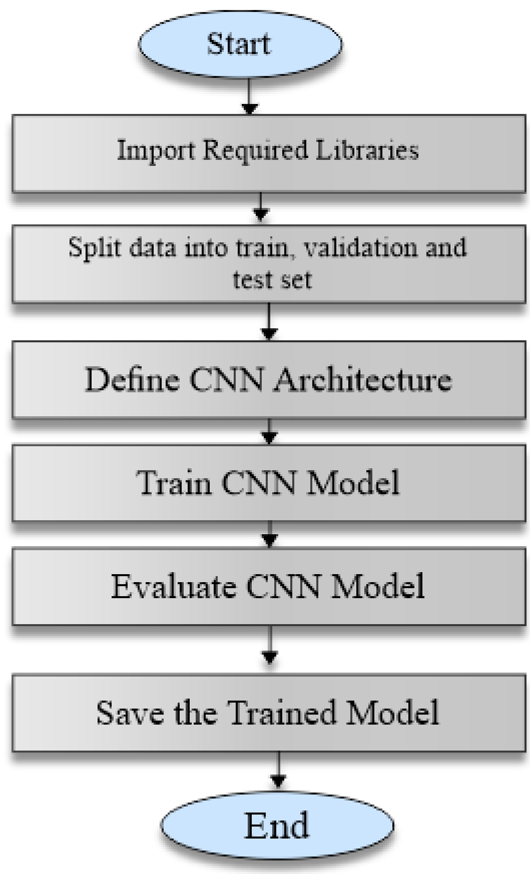

Figure 4.8 CNN Flowchart

- Convolutional neural networks (CNNs) are introduced due to their simplicity and utility in understanding backpropagation equations for convolutional networks. Unlike fully connected neural networks that take vector inputs, CNNs process 2-D array inputs (images).
- While both types of networks perform similar computations—forming a sum of products, applying an activation function, and passing the result to the next layer—CNNs uniquely learn 2-D features from raw image data.

As shown in figure 4.10, which illustrates the convolutional neural network (CNN) architecture with four convolutional layers, each followed by batch normalization and max pooling. The final pooled feature maps are flattened and fed into a fully connected neural network, which produces the output classification.

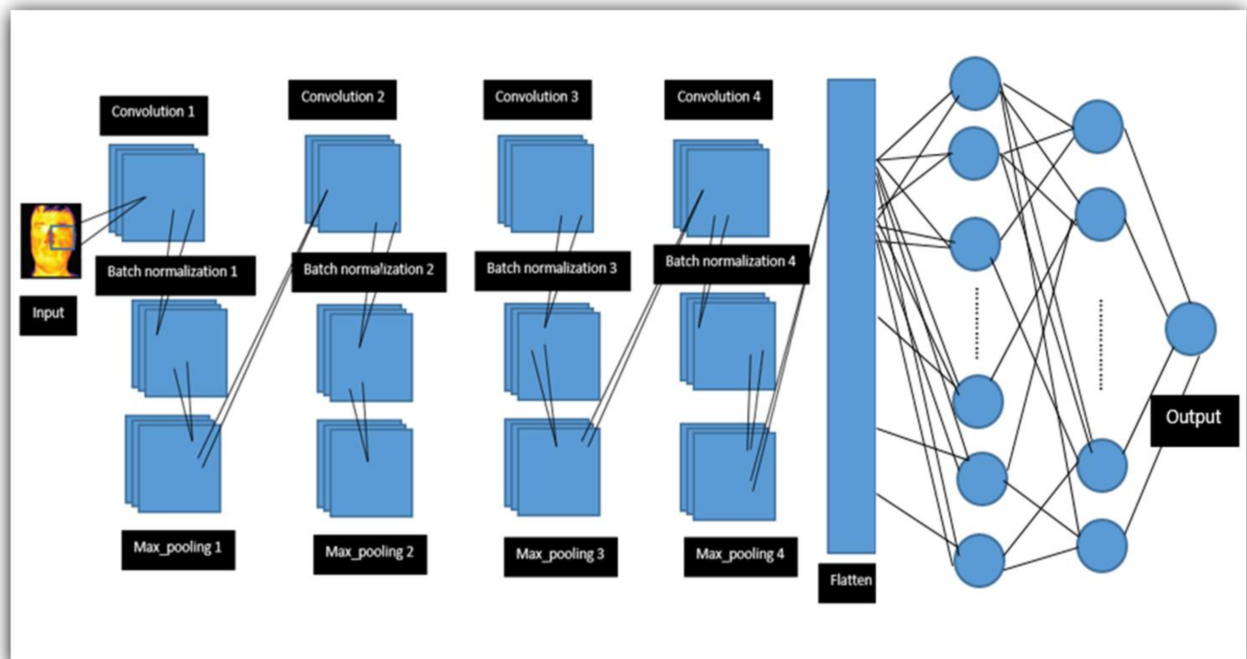

Figure 4.9 CNN Model Architecture

The explanation focuses on a single image input, where the final pooled feature maps are vectorized and input into a fully connected neural network. The class of the input image is determined by the neuron with the highest output value.

Using **Keras** library to define the model architecture we added main Sixteen Layers (4Convolutional, 4 max pooling,4 Batch Normalization, a flatten and 3 dense layers)
As shown in table 4.2:

Table 4.2 CNN Model Full Architecture

Layer Type	Parameters	Activation Function
Convolution	**32 Neuron Kernel Size = (3,3) Input shape = (150,150,3)**	**Relu (Rectified linear unit)**
Batch Normalization	**N/A**	**N/A**
Max Pooling	**Kernel Size = (2,2)**	**N/A**
Convolution	**64 Neuron Kernel Size = (3,3)**	**Relu**
Batch Normalization	**N/A**	**N/A**
Max Pooling	**Kernel Size = (2,2)**	**N/A**
Convolution	**128 Neuron Kernel Size = (3,3)**	**Relu**
Batch Normalization	**N/A**	**N/A**
Max Pooling	**Kernel Size = (2,2)**	**N/A**
Convolution	**256 Neuron Kernel Size = (3,3)**	**Relu**
Batch Normalization	**N/A**	**N/A**
Max Pooling	**Kernel Size = (2,2)**	**N/A**
Flatten	**N/A**	**N/A**
Dense	**256 Neuron**	**Relu**
Dense	**128 Neuron**	**Relu**
Dense	**64**	**Relu**
Dropout	**Rate: 0.2 (20%)**	**N/A**
Dense (Output)	**63 Neuron**	**Softmax**

4.7.2 Training

CNN is trained on the set of images. With total number of parameters 3,647,424 and total number of trainable parameters of 3,646,336.
This is our CNN model hyperparameters that have been trained with:

Table 4.3 CNN Best Hyperparameters

Parameters	CNN
Optimizer	Adam
Loss Function	sparse_categorical_crossentropy
Number of Epochs	70
Batch Size	128

Training Experiments

Our goal was to explore the performance of CNN with different optimization techniques and to identify the best approach for the model. To accomplish this, we designed several experiments using different combinations of hyperparameters. By experimenting with a variety of approaches, we hoped to gain insight into how different factors can impact the accuracy and efficiency of a CNN model, and ultimately improve our ability to classify individuals accurately and efficiently.

Table 4.4 CNN Training Experiment

Experiment No.	Training time	parameters	Accuracy
EX1	39 min	Batch size = 32 No of epoch= 70 optimizer= Adam	90.56%
EX2	25 min	Batch size = 256 No of epoch= 100 optimizer=RmsProp	85.65%
EX3	20 min	Batch size = 64 No of epoch= 70 optimizer= RmsProp	89.60%
EX4	10 min	Batch size = 64 No of epoch= 30 optimizer= Adam	65.73%
EX5	18 min	Batch size = 128 No of epoch= 100 optimizer= Adam	92.46%
EX6	13 min	Batch size = 128 No of epoch= 70 optimizer= Adam	97.46%

These experiments were conducted on a cloud machine, so training time and performance were impacted during the training process.
local machine specifications:

- Processor: Intel Xeon Platinum 8272CL (3.9GHz)
- RAM: 29 GB
- GPU: T4x2

4.7.3 Classification

The final layer of the CNN is a classification layer. The classification layer classifies the input features into one of 63 classes. The classification layer is typically a SoftMax layer, which outputs a probability distribution over the 63 classes in a dense layer with 63 units. This layer takes the flattened output from the previous layers and applies a set of weights to produce an output vector of size 63, which represents the probabilities of the input belonging to each of the 63 classes.

Model Summary

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 148, 148, 32)	896
batch_normalization (BatchNormalization)	(None, 148, 148, 32)	128
max_pooling2d (MaxPooling2D)	(None, 74, 74, 32)	0
conv2d_1 (Conv2D)	(None, 72, 72, 64)	18,496
batch_normalization_1 (BatchNormalization)	(None, 72, 72, 64)	256
max_pooling2d_1 (MaxPooling2D)	(None, 36, 36, 64)	0
conv2d_2 (Conv2D)	(None, 34, 34, 128)	73,856
batch_normalization_2 (BatchNormalization)	(None, 34, 34, 128)	512
max_pooling2d_2 (MaxPooling2D)	(None, 17, 17, 128)	0
conv2d_3 (Conv2D)	(None, 15, 15, 256)	295,168
batch_normalization_3 (BatchNormalization)	(None, 15, 15, 256)	1,024
max_pooling2d_3 (MaxPooling2D)	(None, 7, 7, 256)	0
flatten (Flatten)	(None, 12544)	0
dense (Dense)	(None, 256)	3,211,520
dense_1 (Dense)	(None, 128)	32,896
dense_2 (Dense)	(None, 64)	8,256
batch_normalization_4 (BatchNormalization)	(None, 64)	256
dropout (Dropout)	(None, 64)	0
dense_3 (Dense)	(None, 64)	4,160

Figure 4.10 CNN Model Summary

4.7.4 Model’s Performance Evaluation

The graphs illustrate the performance of a machine learning model during training and validation over 70 epochs.

In Fig 4.12, the model's accuracy rapidly increases during the initial epochs and then stabilizes at a high level (near 1 for training and around 0.98 for validation), indicating effective learning and good generalization to unseen data.

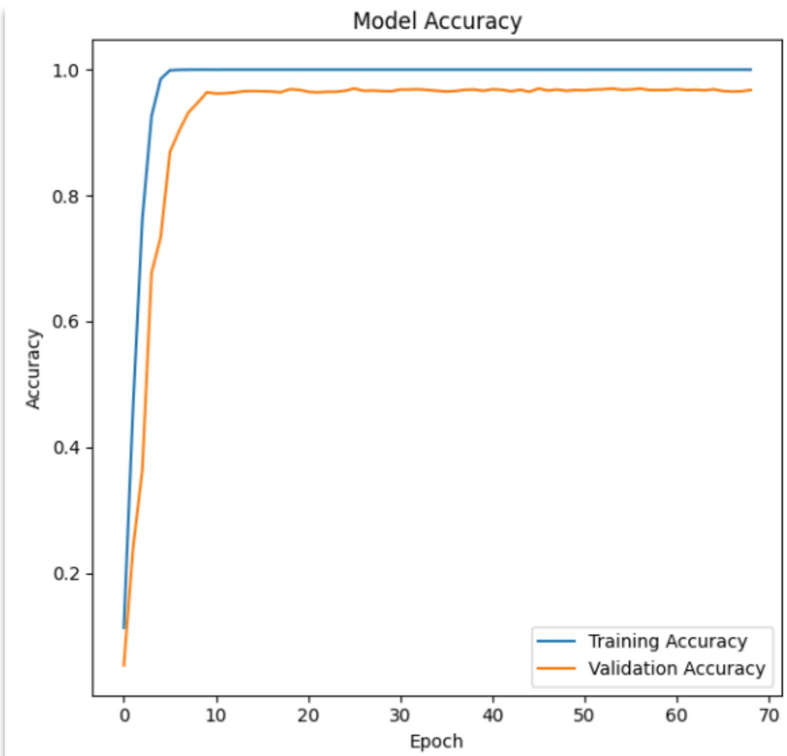

Figure 4.11 CNN Accuracy Curve

In Fig 4.13, the model's loss decreases rapidly in the initial epochs and stabilizes at a low value (near 0 for training and slightly higher for validation), indicating the model effectively minimizes errors on both training and validation datasets.

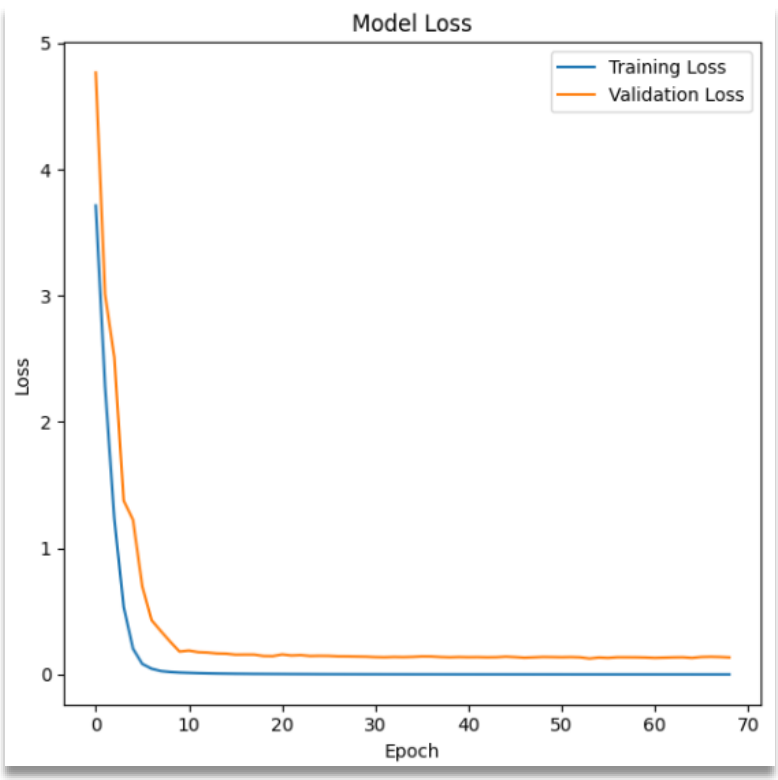

Figure 4.12 CNN Loss curve

Overall, the model achieves high accuracy and low loss, demonstrating strong performance and minimal overfitting.

4.8 GUI

4.8.1 Home page:

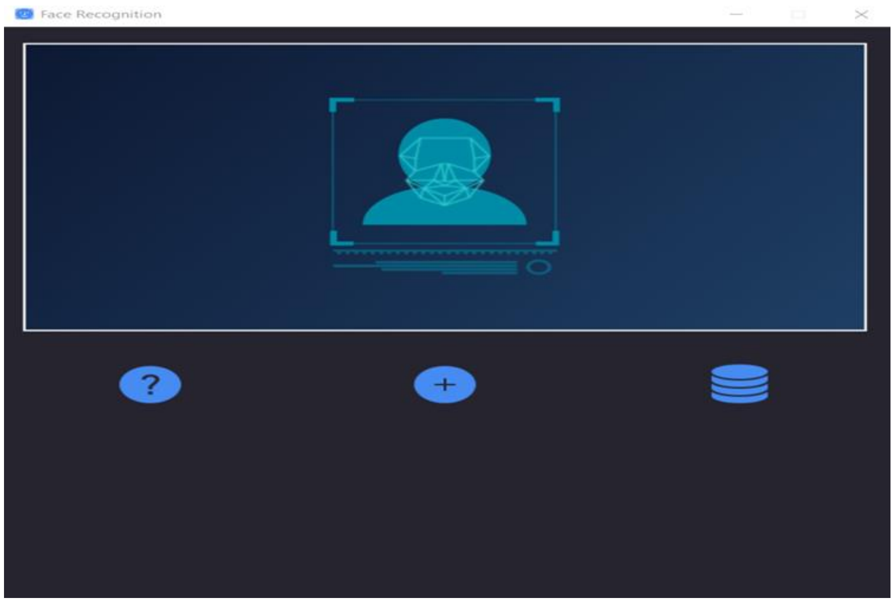

Figure 4.13 Home page

The home page of the face recognition application serves as the main interface, providing access to various functionalities. It features three buttons:

Help Button (Question Mark): Opens a help menu or documentation to guide the user on how to use the application and troubleshoot issues.

Start Recognition Button (Plus Sign): Initiates the face recognition process, allowing the user to upload or capture images for identification.

History Button (Stack of Discs): Provides access to the history of recognized persons, displaying details such as names, dates, and times of recognition.

4.8.2 Start page:

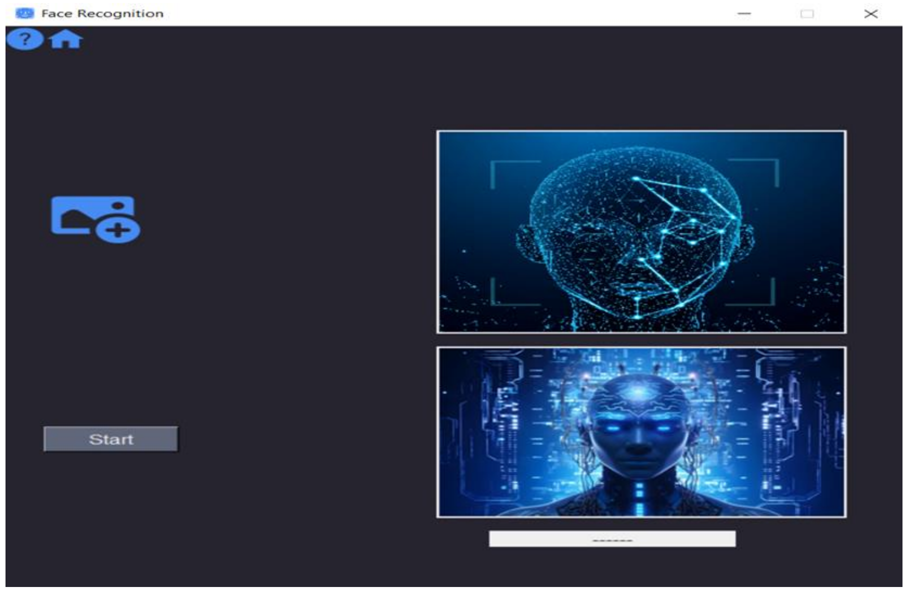

Figure 4.14 Start page

the start page for a face recognition application. It serves as the initial interface where users can begin the process of face recognition.

First Button (Top left with a home icon): Takes the user back to the home page.

Second Button (Below the home icon, with an image icon): Allows the user to add an image for recognition.

The third Button (Bottom left, labeled "Start"): Starts the face recognition process.

Fourth Button (Top right, with a question mark icon): Provides help or guidance about using the application.

4.8.3 Person's identity:

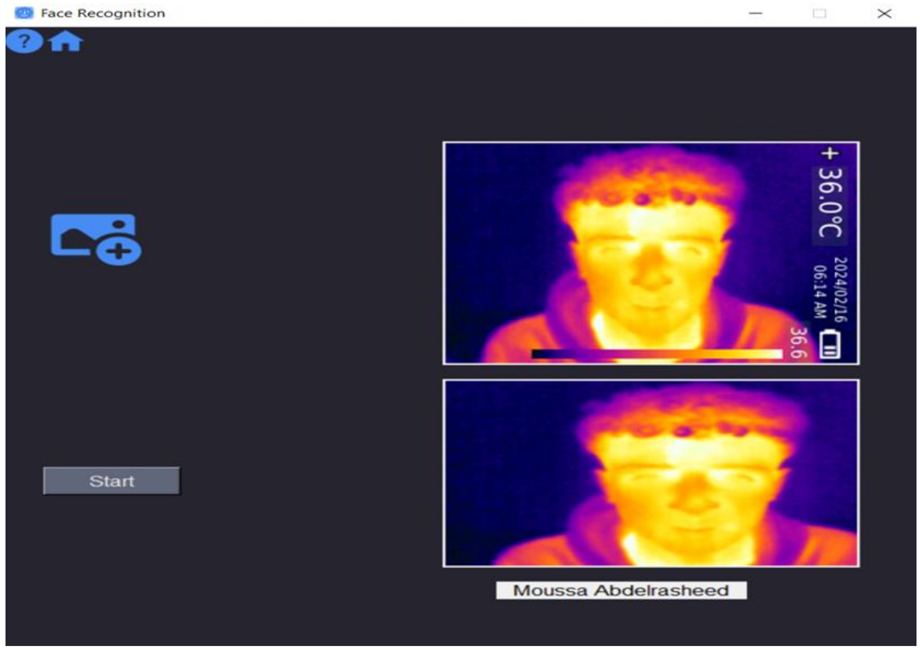

Figure 4.15 Person identity

The start page after the recognition process has been completed and the identity of the person appears under the preprocessed image.

4.8.4 Photo Missing Error Handling:

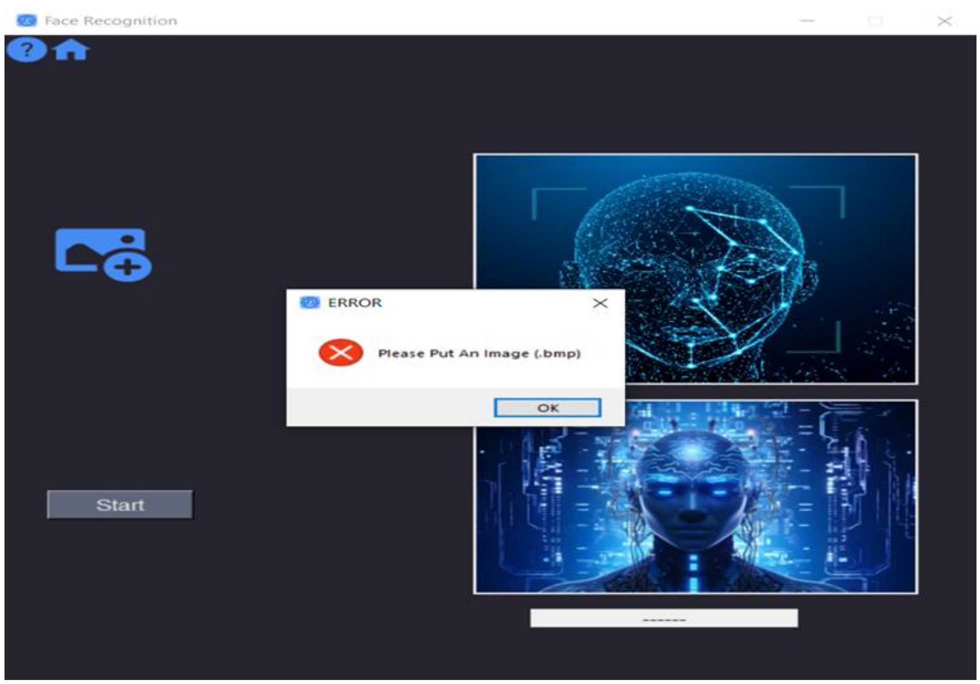

Figure 4.16 Photo missing error handling

4.8.5 History Page:

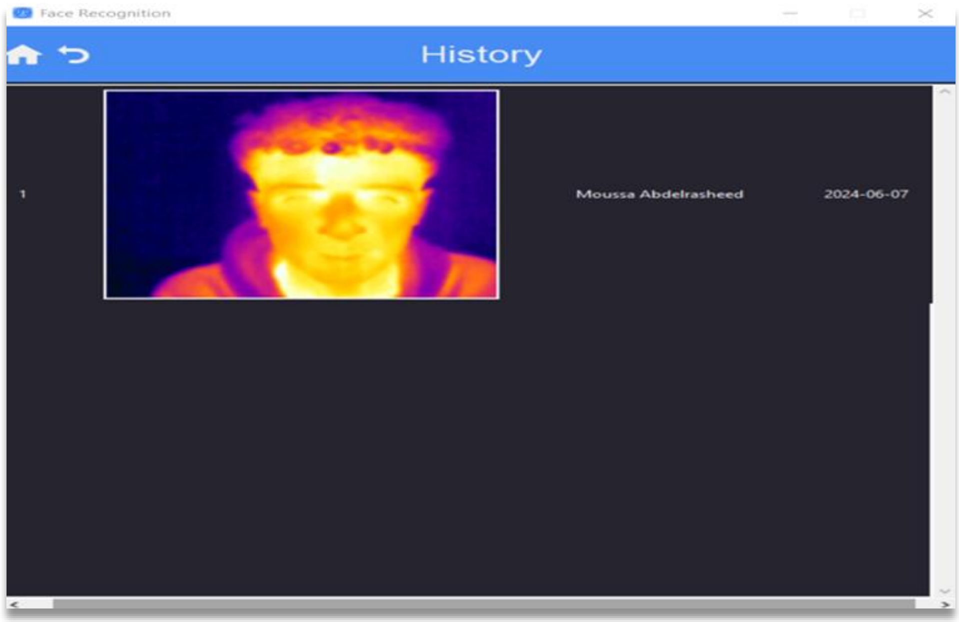

Figure 4.17 History page

The history page in the face recognition application records and displays previously recognized faces, along with the date and time of recognition. It helps users verify and review past recognitions for accuracy and tracking purposes. The page shows an infrared image of each recognized person, their name, and the recognition date.

Chapter Five

Testing

Chapter Five Testing

5.1 Testing

Software testing is the process of assessing the functionality of a software program. The process checks for errors and gaps and whether the outcome of the application matches desired expectations before the software is installed and goes live.

What is the purpose of the test case?

The main purpose of a test case is to verify if a software application functions correctly and meets the requirements that were laid out for it. Test cases achieve this by outlining specific actions, input values and expected outcomes [1]. In essence, they are a way to check if the system behaves as intended under various conditions, both normal and abnormal.

Test case:

1- The user tries to know how can use our application

Table 5.1 Test Case 1

Test case	**Scenario**	**Test step**	**Expected result**	**Actual outcome**
TC-001	The user tries to click the help icon to learn some information about our application	Navigate to the dashboard page and click on the first icon (help icon)	Any images or buttons on the application should be clear and correctly displayed.	It displays the functionality of buttons on the dashboard page.

2- The user tries to select a face image on his/her desktop or our camera

Table 5.2 Test Case 2

Test case	**Scenario**	**Test step**	**Expected result**	**Actual outcome**
TC-002	The user tries to click the Add icon to select his/her face image	Click on the add button to select an image from the user's desktop.	The user should be able to browse and select an image file from their desktop	Once an image is selected, it should be displayed in the first dynamic image area on the selected page.

3- The user tries to apply recognition on the selected image

Table 5.3 Test Case 3

Test case	**Scenario**	**Test step**	**Expected result**	**Actual outcome**
TC-003	The user tries to click on the add icon then click add an image to select the image and click on the start button	Click on the start button to perform recognition	The face recognition process should start immediately after the button is clicked.	The second dynamic image area should be cleared and replaced by the user's recognized image

4-User tries to check the name of the recognized image

Table 5.4 Test Case 4

Test case	**Scenario**	**Test step**	**Expected result**	**Actual outcome**
TC-004	The user tries to click on the history icon	Check that the text box displays the name of the recognition.	The name should be clearly visible and correctly spelled.	The recognition name should be updated in real-time as the face recognition process concludes.

5-User tries to open the history page

Table 5.5 Test Case 5

Test case	**Scenario**	**Test step**	**Expected result**	**Actual outcome**
TC-005	The user tries to access the history page	Click on the history button to show past recognitions	The history page should display a list of all previous recognition events.	Each entry in the history should include the date and time of recognition, the name of the recognized individual, and a thumbnail of the recognized image.

5.2 Result after recognition

Figure 5.1 Actual output of the program

With recognition accuracy of 97.5% of the input image.

Chapter Six

Conclusion & Future Work

Chapter Six Conclusion & Future Work

6.1 Conclusion

In conclusion, this documentation has comprehensively explored the application of convolutional neural networks (CNN) for face recognition using thermal images:

- **Exploration of CNN for Face Recognition:**
  - Comprehensive documentation on the use of CNNs for face recognition using thermal images.
  - Thermal imaging offers substantial advantages, particularly in low-light environments and situations where visible spectrum imaging is hindered.

- **Benefits of Thermal Imaging:**
  - Effective in low-light or completely dark environments.
  - Beneficial for surveillance and security applications with uncontrolled lighting conditions.
  - Less susceptible to ambient lighting variations, ensuring consistent performance.
  - Can penetrate disguises like makeup or camouflage, enhancing security.

- **Efficacy of CNNs:**
  - CNNs excel in pattern recognition and image classification.
  - Superior accuracy of CNNs in recognizing faces from thermal images compared to traditional methods.
  - Robust handling of variations in facial expressions, head poses, and partial occlusions.

- **Extensive Testing:**
  - Used diverse datasets to simulate real-world scenarios.
  - Included a wide range of subjects and environmental conditions.
  - Controlled experiments evaluated the model's recognition ability with different expressions and partial occlusions.

- **Importance of Dataset Quality and Future Research:**
  - Dataset quality and diversity are crucial for improving model generalization.
  - Aim to boost performance and explore new possibilities for real-time applications.

So overall, the integration of thermal imaging and CNNs in face recognition technology offers promising applications in security, surveillance, and biometric authentication. This research provides a strong foundation for future advancements, aiming to develop more sophisticated and resilient systems. Continuous testing and improvements will enhance accuracy and applicability to meet industry demands. Notably, the developed thermal face recognition system achieved a 97.4% accuracy rate with a 3-second response time.

6.2 Future Work

The current project has successfully demonstrated the use of convolutional neural networks (CNNs) for face recognition using thermal images, achieving promising results with a dataset of 63 individuals and incorporating a user interface. To further enhance the system's performance, robustness, and applicability, the following areas are proposed for future exploration and development:

-Robustness and Imperceptibility Improvements

Enhancing the robustness and imperceptibility of the face recognition system is crucial for ensuring its reliability and effectiveness in real-world applications. The following strategies can be employed to achieve these improvements:

6.2.1 Expansion of Dataset

To enhance the performance and robustness of the face recognition system, expanding the dataset is imperative. Incorporating a larger dataset with diverse facial expressions, orientations, and environmental conditions can facilitate better training of the convolutional neural network (CNN). Additionally, including thermal images captured under various lighting conditions and temperatures can improve the model's adaptability to real-world scenarios.

6.2.2 Fusion of Modalities

Consider integrating additional modalities, such as visible light images or depth information, with thermal images to create a multimodal face recognition system. Fusion of complementary modalities can mitigate the limitations of individual modalities and improve the overall recognition performance. Explore different fusion strategies, including early fusion at the input level or late fusion at the feature level, to determine the most effective approach for combining modalities.

GANs: A CNN trained on a limited dataset might struggle with recognizing faces in unseen scenarios. Fusing a GAN with CNN can address this. The GAN can generate synthetic faces with variations in pose, lighting, or occlusions. This enriched dataset can improve CNN's ability to handle real-world complexities, boosting recognition accuracy.

Transformers for Feature Extraction: While CNNs excel at extracting spatial features, they might struggle with long-range dependencies in facial features. Transformers, with their "attention" mechanism, can focus on crucial relationships between distant facial regions like eyes, nose, and mouth. Fusing a Transformer with a CNN allows the model to leverage both local and global features, potentially leading to more robust face recognition.

6.2.3 Real-Time Implementation and Optimization

Transition the face recognition system from a graphical user interface (GUI) to a real-time implementation capable of processing thermal images in live video streams.

References

[1] “National Institute of Standards and Technology (NIST)” (2023)

[2] Rafael C. Gonzalez, Richard E. Woods “Digital Image Processing: FOURTH EDITION” (2019)

[3] Michael Blumenstein “Pattern Recognition Technologies and Applications Recent Advances” (2019)

[4] Xuerui Dai, Xue Yuan “Object detection in thermal infrared images for autonomous driving” (2021)

[5] SP Mukherjee “A Guide to Research Methodology: An Overview of Research Problems, Tasks and Methods” (2020)

[6] Christopher Pramerdorfer, Martin Kampel “Face Expression Recognition using CNN” (2021)

[7] D. T. Nguyen, J. Kang “Thermal Image Reconstruction Using Deep Learning” (2019)

[8] Delong Qi, Xiangyu Li “YOLO5Face: Why Reinventing a Face Detector” (2022)

[9] Waseem Rawat, Zenghui Wang “Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review” (2021)

[10] Darina Koishigarina, Askat Kuzdeuov (Member, IEEE) “TFW: Annotated Thermal Faces in the Wild Dataset” (2022)

[11] Mohamed Sayed, Faris Baker “Thermal Face Authentication with Convolutional Neural Network” (2020)

[12] Yue Zheng, Mingzhong Yi “Multiscale Fusion of Visible and Thermal IR Images for Illumination-Invariant Face Recognition” (2019)

[13] Mais Muneam, Mohd Dilshad “Local Feature Methods Based Facial Recognition” (2022)