UNIVERSITY OF KWAZULU-NATAL

COLLEGE OF AGRICULTURE, ENGINEERING AND SCIENCE

COMBINING LOCAL DESCRIPTORS AND CLASSIFICATION METHODS FOR HUMAN EMOTION RECOGNITION

by

Antoine Badi Mame

217009138

Supervisor:

Prof. Jules-Raymond Tapamo

In fulfillment of the academic requirements of the degree of Master of Science in Engineering, School of Engineering, University of KwaZulu-Natal

March 31, 2023

PREFACE

The research described in this thesis was performed at the University of KwaZulu-Natal under the supervision of Prof. Jules-Raymond Tapamo. I, hereby, declare that all materials incorporated in this thesis are my own original work except where the acknowledgement is made by name or in the form of reference. The work contained herein has not been submitted in part or whole for a degree at any other university.

Antoine Badi Mame

As the candidate's Supervisor, I agree to the submission of this dissertation.

[redacted]

Jules-Raymond Tapamo

COLLEGE OF AGRICULTURE, ENGINEERING AND SCIENCE

DECLARATION 1 - PLAGIARISM

I, Antoine Badi Mame, declare that

1. The research reported in this dissertation, except where otherwise indicated, is my original research.
2. This thesis has not been submitted for any degree or examination at any other university.
3. This thesis does not contain other persons' data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.
4. This thesis does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:
   (a) Their words have been re-written but the general information attributed to them has been referenced
   (b) Where their exact words have been used, then their writing has been placed in italics and inside quotation marks, and referenced.
5. This thesis does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the thesis and in the References sections.

Antoine Badi Mame

COLLEGE OF AGRICULTURE, ENGINEERING AND SCIENCE

DECLARATION 2 – PUBLICATIONS

Details of contribution to publications that form part and/or include research presented in this dissertation:

1. A. Badi Mame and J.-R. Tapamo, “A Comparative Study of Local Descriptors and Classifiers for Facial Expression Recognition,” Applied Sciences, vol. 12, no. 23, p. 12156, Nov. 2022, doi: 10.3390/app122312156.
2. A. Badi Mame and J. R. Tapamo, “Parameter Optimization of Histogram-based Local Descriptors for Facial Expression Recognition,” PeerJ Computer Science (in press).

Antoine Badi Mame

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my Lord and Saviour, Jesus-Christ, for giving me the life and grace to study for this degree.

I would like to thank my supervisor, Prof. Jules-Raymond Tapamo for his guidance and mentorship which have taught me invaluable lessons that I will never forget.

I would like to thank my family and friends. Without their prayers, encouragement, and support, I would not have made it this far.

Antoine Badi Mame

ABSTRACT

Human Emotion Recognition occupies a very important place in artificial intelligence and has several applications, such as emotionally intelligent robots, driver fatigue monitoring, mood prediction, and many others. Facial Expression Recognition (FER) systems can recognize human emotions by extracting face image features and classifying them as one of several prototypic emotions. Local descriptors are good at encoding micro-patterns and capturing their distribution in a sub-region of an image. Moreover, dividing the face into sub-regions introduces information about micro-pattern locations, essential for developing robust facial expression features. Hence, local descriptors' efficiencies depend heavily on parameters such as the sub-region size and histogram length. However, the extraction parameters are seldom optimized in existing approaches.

This dissertation reviews several local descriptors and classifiers, and experiments are conducted to improve the robustness and accuracy of existing FER methods. A study of the Histogram of Oriented Gradients (HOG) descriptor inspires this research to propose a new face registration algorithm. The approach uses contrast-limited histogram equalization to enhance the image, followed by binary thresholding and blob detection operations to rotate the face upright. Additionally, this research proposes a new method for optimized FER. The main idea behind the approach is to optimize the calculation of feature vectors by varying the extraction parameter values, producing several feature sets. The best extraction parameter values are selected by evaluating the classification performances of each feature set. The proposed approach is also implemented using different combinations of local descriptors and classification methods under the same experimental conditions.

The results reveal that the proposed methods produced a better performance than what was reported in previous studies. Furthermore, the results showed an improvement of up to 2% compared with the performance achieved in previous works. The results showed that HOG was the most effective local descriptor, while Support Vector Machines (SVM) and Multi-Layer Perceptron (MLP) were the best classifiers. Hence, the best combinations were HOG+SVM and HOG+MLP.

Keywords: Facial Expression Recognition; Local descriptors; Classification; Feature extraction.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF ABBREVIATIONS

AAM	Active Appearance Model
Adam	Adaptive moment estimation
ALDP	Angled Local Directional Patterns
AUC	Area Under the Curve
BNN	Binarized Neural Networks
BPPC	Binary Pattern of Phase Congruency
CART	Classification and Regression Tree
CBP	Centralized Binary Pattern
CK	Cohn Kanade
CK+	Extended Cohn Kanade
CLBP	Compound Local Binary Patterns
CLGDNP	Compact Local Gabor Directional Number Pattern
CNN	Convolutional Neural Networks
DBN	Deep Belief Network
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
dLHD	directed Line segment Hausdorff Distance
DRADAP	Decoder Regional Adaptive Affinitive Patterns
ELM	Extreme Learning Machine
ELBPTOP	Extended Local Binary Patterns on Three Orthogonal planes
FACS	Facial Action Coding System
FER	Facial Expression Recognition
FRR	Feature Redundancy-Reduced
GDP	Gradient Directional Pattern
GLTeP	Gradient-based Local Ternary Patterns
HCRF	Hidden Conditional Random Fields
HMM	Hidden Markov Models
HOG	Histogram of Oriented Gradients
JAFFE	Japanese Female Facial Expression

KNN	K-Nearest Neighbor
LAP	Local Arc Pattern
LBP	Local Binary Patterns
LBPTOP	Local Binary Pattern on Three Orthogonal Planes
LDA	Linear Discriminant Analysis
LDBP	Local Dominant Binary Pattern
LDFS	Local Direction Feature Structure
LDN	Local Directional Number
LDP	Local Directional Patterns
LDPv	Local Directional Pattern with Variance
LGCP	Local Gray Code Pattern
LGDiP	Local Gabor Directional Pattern
LGIP	Local Gradient Increasing Pattern
LGP	Local Gradient Pattern
LGTrP	Local Gabor Transitional Pattern
LLE	Locally Linear Embedding
LMP	Local Monotonic Pattern
LOSO	Leave-One-Subject-Out
LPDP	Local Prominent Directional Pattern
LSTM	Long Short-Term Memory
LTP	Local Ternary Patterns
LTrP	Local Transitional Pattern
LVQ	Learning Vector Quantization
MDC	Minimum Distance Classifier
MFFNN	Multilayer Feed Forward Neural Network
MLP	Multi-Layer Perceptron
MRDNP	Maximum Response-based Directional Number Pattern
MRDTP	Maximum Response-based Directional Texture Pattern
MBP	Median Binary Pattern
MUG	Multimedia Understanding Group

NB	Naive Bayes
PC	Phase Congruency
PCA	Principal Component Analysis
PFI	Perceived Facial Images
PHOG	Pyramid Histogram of Oriented Gradients
PZM	Pseudo Zernike Moments
RBF	Radial Basis Function
RFD	RadBoud Faces Database
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristics
SVM	Support Vector Machines
ULTP	Uniform Local Ternary Pattern
WLD	Weber’s Local Descriptor
WSBP	Weighted Statistical Binary Pattern

CHAPTER 1
INTRODUCTION

This dissertation is about Human Emotion Recognition based on analyzing face images in computer systems.

1.1 Background

Emotion plays a central role in human interactions with other humans and human-machine interactions. To make computers more useful and accessible to humans, it is essential to understand their feelings and behavior in response to various stimuli [1]. For example, service robots demonstrating emotional awareness could make human-computer interactions more natural and enjoyable. Marketing strategies that develop specialized adverts and customer reviews focused on users' emotional states will likely increase product acceptance [2]. Research in Human Emotion Recognition has been used to develop assistive technology for disabled people [3]. It is also used in psychological studies for diagnosing mental diseases and concerning the level of technology adoption and effectiveness of remote learning and working [4].

Several theories and models of emotion have been proposed to classify the spectrum of human emotions. Feidakis et al. [5] identified up to 66 emotions and categorized them as basic/primary (e.g., anger, anticipation, distrust, and fear) and secondary emotions (e.g., hopelessness, envy, and satisfaction). However, the overlapping nature, subjectivity, and the sheer number of emotions make it challenging to develop systems to recognize human emotions. For this reason, most studies have focused on basic emotions, which are more easily defined. Russel's [6] circumplex model of emotions distributes the basic emotions on a two-dimensional space with respect to the valence (whether negative or positive) and arousal (i.e., the intensity) components of emotion (see Figure 1.1).

Emotion recognition methods can be differentiated based on which aspects of the human body are used to detect emotional states. Techniques based on electroencephalography recognize emotions by monitoring the brain's electrical activity, while electrocardiography measures the heart's electri-

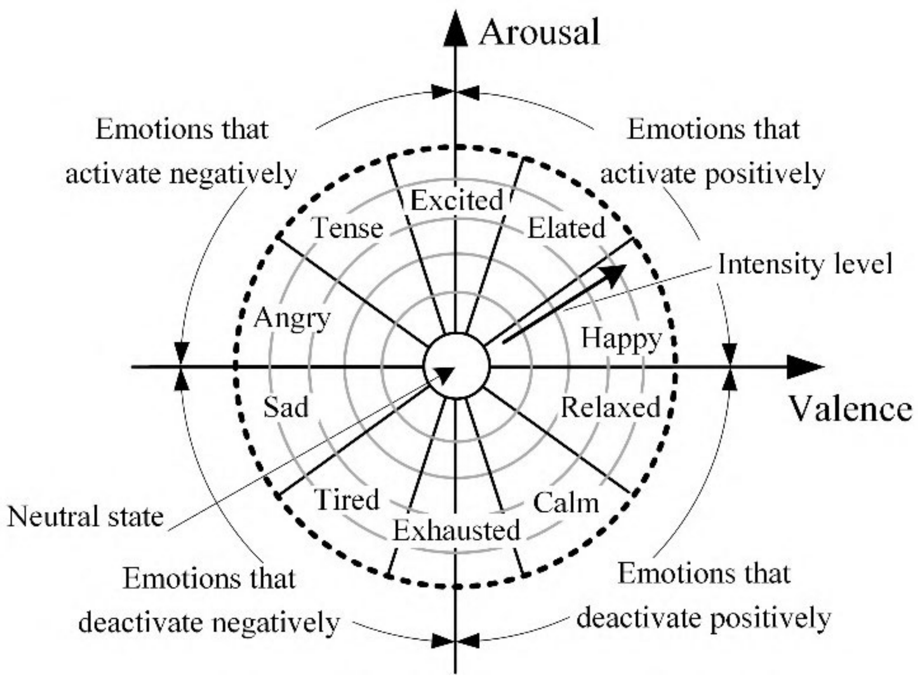

Arousal
Valence
Emotions that activate negatively
Emotions that activate positively
Emotions that deactivate negatively
Emotions that deactivate positively
Intensity level
Neutral state
Excited
Elated
Happy
Relaxed
Calm
Exhausted
Tired
Sad
Angry
Tense

Figure 1.1: Russel's circumplex model of emotions [1].

cal activity. Other methods include skin resistance measurements, blood pressure, eye activity, body gestures, and facial expressions. Recent advances in computer vision have increased interest in developing emotion recognition systems based on facial expressions. Facial Expression Recognition (FER) does not require contact with the subject and provides high accuracy and reliability. However, FER systems are challenged by several issues stemming from face image acquisition. Mainly, camera sensors are used; hence, factors such as image resolution, scene illumination, head pose, and occlusions are some difficulties that only FER systems face.

Three processes are essential when developing a robust FER system: 1) Face image acquisition, 2) Feature extraction, and 3) Classification. Face detection aims at locating a face region and returning its position. The human face contains much information, which is not necessarily helpful in recognizing facial expressions. Feature extraction is essential for obtaining an appropriate face representation that characterizes face deformations while discarding non-expression information. The facial features must then be classified as one of the prototypic emotions.

Local descriptors are a collection of image features that extract textural lines, spots, edges, and corners from images. Several approaches have been developed based on local descriptors due to their simplicity and efficiency in characterizing facial expressions [7, 8, 9]. Deep learning techniques have also attracted much attention from the community due to their increased capacity for high recognition rates [10]. The debate between handcrafted and deep features is still active because it is challenging to obtain robust facial expression features [11].

1.2 Motivation and Problem Definition

A FER system typically operates in three stages: in the first stage, a face region is detected; in the second stage, facial features are extracted to produce a feature vector; and finally, the feature vector is classified as a facial expression in the third stage. Although much research has been done in this area, many approaches are still under-underutilized and can be improved. After a face is detected, a common task is normalizing the face region by performing operations such as contrast enhancement, cropping, resizing, and face alignment. In [12], this process is called face registration, where Viola-Jones-based algorithms and ellipse fitting techniques are used to detect a face region, align it and remove background objects. However, this method relies on facial feature color models to rotate the face blob, limiting its applicability to gray-scale images. Although face registration is part of many FER approaches, little effort has been made to formulate an automated solution. This motivates this study to propose an automated face registration algorithm. Additionally, the impact of face registration on the recognition performance of various FER approaches must be investigated.

Feature extraction is aimed at generating a feature vector from a face image. The extracted features must be robust to illumination variation, pose changes, occlusions, and low-resolution images. Local descriptors are a powerful way of representing expression-related information. In particular, the histogram-based local descriptors represent the occurrences of micro-patterns that make up facial expressions. It is known that the aggregation of these micro-patterns into histograms leads to a loss of information about the location of the micro-patterns [13, 14]. One solution is dividing the face region into overlapping or non-overlapping sub-regions. Shan et al. [13] used the Local Binary Patterns (LBP) descriptor to extract features from sub-regions of a face image. After that, a histogram was created with 59 bins and extracted from 18×21 pixels sub-regions in a 110×150 image. However, the extraction parameters (i.e., number of bins and sub-region) seemed arbitrarily chosen. Carcagni et al. [12] studied the Histogram of Oriented Gradients (HOG) descriptor and proposed a method for optimizing the cell size and histogram bins during the feature extraction. The study compared the HOG descriptor with three other local descriptors. However, the method for selecting the other descriptors' parameters was not given. This motivates an investigation into optimizing the histogram-based local descriptors for Facial Expression Recognition.

The final task of FER is to label a feature vector as one of a set of prototypic expressions. Machine learning classification methods help perform this task by creating a model for labeling feature vectors based on a dataset. Some classification methods, such as Support Vector Machines (SVM), are designed to divide the feature space into discrete classes. Recent studies have attempted to review various local descriptors for Facial Expression Recognition. However, most of the time, only one or two classifiers are used [11, 15]. Furthermore, whenever a method is evaluated, it is often compared to other studies conducted under different conditions. Hence, it is necessary to analyze and compare the performance of different combinations of local descriptors and classification methods under the same experimental conditions.

1.3 Main Goal and Specific Objectives

The main goal of this research is to investigate existing local descriptors and classification techniques for Human Emotion Recognition. This will ultimately require developing Facial Expression Recognition systems based on facial images.

Developing a robust and reliable FER system is particularly challenging because several factors must be considered, such as the camera's view angle, obstructing objects (or occlusions), the image resolution, the scene illumination, and the color channels considered. The selection of which factors to prioritize is highly dependent on the application. For example, service robots and surveillance cameras must be able to handle different head poses and low image resolutions. On the other hand, online learning and video meetings are most likely to deal with frontal-view images, higher image resolutions, and few view obstructions. This research considers frontal face images and low-resolution images as challenging conditions for FER. The algorithms were developed to reduce computational costs while optimizing the recognition performance, assuming the input images were in the gray-scale format.

The following specific objectives have been carried out:

- Replicate a study on the Histogram of Oriented Gradients (HOG) descriptor to confirm, understand and verify observations found in theory and literature.

- Obtain datasets with appropriate data that is unbiased of any factor or characteristic through extensive literature surveys.
- Implement selected local descriptors and classifiers, depicting a comparative contrast between the distinct methodologies and their suitability contexts and problems.
- Analyze and discuss the results, portraying strengths, weaknesses, and situational trade-offs among each model and chosen local descriptors.
- Identify the most suitable combination of classifiers and local descriptors for Human Emotion Recognition.

1.4 Contributions

This work contributes to the field of Human Emotion Recognition in the following ways:

- Identified factors that can improve the discriminability and robustness of existing and future local descriptors for Facial Expression Recognition. One of the factors is the development of expression-specific features.
- Proposed a novel face registration algorithm for aligning, cropping, and resizing the face image before extracting features.
- Found that several local descriptors have been underutilized due to the lack of parameter optimization. For instance, when used in its original form, the LBP descriptor is inefficient. However, appropriately selecting the number of histogram bins and sub-region size can improve its classification performance.
- Analyzed the performances of several local descriptors and classifiers in the FER problem. Experiments were conducted under different settings, such as varied extraction parameters, different numbers of expressions, and two well-known datasets. A new texture descriptor, ALDP (Angled Local Directional Pattern), was examined with several classifiers for the first time.

1.5 Dissertation Outline

- Chapter 2 presents a literature review of existing local descriptors for Facial Expression Recognition. It also discusses face detection and classification techniques.
- Chapter 3 presents a study done to replicate a selected work on the HOG descriptor.
- Chapter 4 discusses a method for optimizing four local descriptors for FER.
- Chapter 5 is a comparative study of six local descriptors and four classifiers for FER.
- Chapter 6 concludes this work and gives some ideas for future works.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

The most common framework for Facial Expression Recognition is to extract facial features and classify them as basic emotions such as anger, surprise, happiness, fear, disgust, and sadness. Much of the research has been devoted to developing local descriptors to represent facial deformations robustly and efficiently. Turan and Lam [11] reviewed histogram-based local descriptors for FER. They compared the performances of 27 local descriptors on four facial expression datasets using the same experimental setup. The following FER approach was implemented: 1) face and facial component localization; 2) image resizing and subdivision; 3) Feature extraction; 4) Dimensionality reduction, and 5) classification. Two different cross-validation schemes were used (i.e., Leave-One-Subject-Out and 10-fold). Local Phase Quantization (LPQ) and Local Gabor Binary Pattern Histogram Sequence (LGBPHS) achieved the best overall performances, giving consistent results across most datasets.

Slimani et al. [15] analyzed the performances of 46 LBP variants for Facial Expression Recognition on four datasets. The FER approach was configured as follows: 1) face localization and resizing; 2) feature extraction; 3) feature image subdivision and histogram calculation; 4) classification. The study used the half-half scheme, and the average per-class accuracies were computed. Results show that texture descriptors initially proposed for tasks other than facial emotion recognition can outperform state-of-the-art systems. The best recognition rates achieved were 100%, 98.57%, 95.92%, and 100% for CK, JAFFE, KDEF, and MUG databases, respectively.

The studies above reveal that local texture descriptors have great potential in developing future FER systems. However, very few reviews have exclusively studied local descriptors designed to tackle the FER problem. This review attempts to fill this gap by presenting FER's most recent local descriptors, including their strengths and weaknesses. An attempt is also made to identify specific factors that could enhance the performance of future FER systems.

As seen in Figure 2.1, the general process for Facial Expression Recognition can be summarized in four steps:

1. **Face acquisition:**

   An input image is searched for a face region whose location must be returned. Additionally, pre-processing operations may be performed to normalize facial appearance.

2. **Feature extraction:**

   Features are extracted by applying a local descriptor (e.g., LBP or HOG) to the registered face. This produces a feature image containing descriptor values in place of pixel intensity values.

3. **Image subdivision and histogram computation:**

   The feature image is divided into equally sized sub-regions. A 1D histogram is computed for each of the sub-regions. An additional step involves normalizing the histogram to increase its generalizability. The normalized histograms of all the sub-regions are concatenated and the final histogram is treated as a feature vector for classification.

4. **Classification:**

   In the training stage, a model is learned based on feature vectors extracted from the training set and their corresponding labels. The learned model is then used to classify a new sample's feature vector as one of the prototypic expressions.

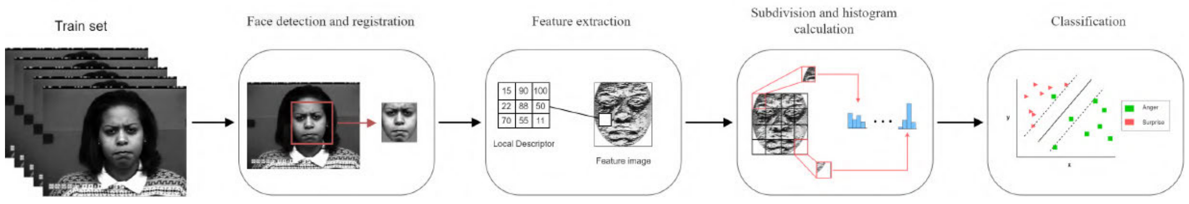


Figure 2.1: Pipeline for a Facial Expression Recognition system using local descriptors.

2.2 Face Acquisition

2.2.1 Overview

Face acquisition is the preliminary step to recognizing a facial expression. The main processes involved in acquiring facial data include face detection and face registration. Face detection is the process of searching an image for a face region. Once the face region has been located, its position and bounds are returned. Face detection is usually followed by face registration/pre-processing, which gives the face region a predefined shape, size, and pose.

2.2.2 Face Detection

Several techniques have been developed for detecting a face region from an image. Some face detection techniques include but are not limited to eigenspace methods, Haar classifiers, AdaBoost classifiers, contour points, and skin color segmentation [16]. The Viola-Jones face detector [17] is one of the most popularly used in modern FER systems [18, 19, 20, 21].

A Viola-Jones face detector is based on extracting Haar-like (i.e., rectangular) features computed from an integral image (allowing features to be computed rapidly) and building a "strong" classifier by combining many "weak" classifiers with increasing complexities. The cascade structure of classifiers allows the background regions to be discarded from the image quickly and the face regions to be found more quickly. Additionally, the framework is resilient to illumination variation, scale variation and has a low computational cost and high true positive rate [12]. Figure 2.2 illustrates the detection of faces of varying resolutions on the MIT + CMU dataset [17]. Some limitations of this strategy are the long training periods and poor performance with rotated face images.

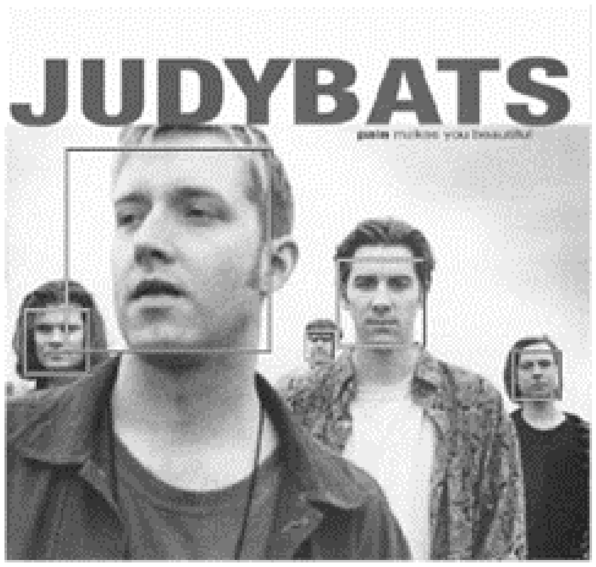


Figure 2.2: Examples of detected faces on the MIT+CMU dataset [17].

2.2.3 Face Registration

Face registration (or face normalization or pre-processing) is essential to discard noise and person-specific information (such as background and hair) from the face image. To train a machine learning model, the input feature vector is required to have a predefined dimensionality. Thus, the basic requirement of every FER system is that all face samples are of the same size, ensuring that all feature vectors have the same dimensionality. In [22], faces are detected using the Dlib library [23], followed by detecting 68 facial landmark points using the facial landmark detector in the same library. The landmark detector uses a regression tree model, which finds these facial points directly from the pixel intensities, allowing for a speedy detection. Finally, the faces are resized to 130×130 pixels. Note that the faces are not aligned. This process is illustrated in Figure 2.3.

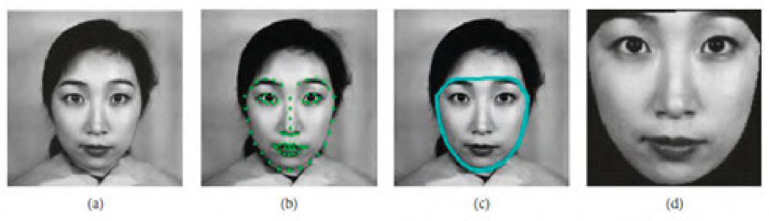


Figure 2.3: A framework for face detection and face extraction: (a) the original image, (b) the detected face with 68 landmarks, (c) the face region, (d) the extracted face. [22].

In [8], 48 facial landmarks are detected using the Cascade Linear Regression (CLR) method. Then, the distance between the eye centers is fixed to 55 pixels, followed by face alignment to reduce head position variability. Finally, the distance between the eyes is used as a measure to resize the image to 110×150 pixels. In [19], the face is detected using the Viola-Jones algorithm and cropped to 100×100 pixels. No other processing is performed. In [12], an ellipse is fitted around the detected face blob to rotate the face to a vertical position. Then, the locations of the eyes provide a measure to crop and scale the face image to a size of 65×59 pixels. Note that the ellipse-fitting algorithm depends on facial feature color models, making it unusable for grayscale images. Figure 2.4 illustrates the face registration algorithm developed in [12].

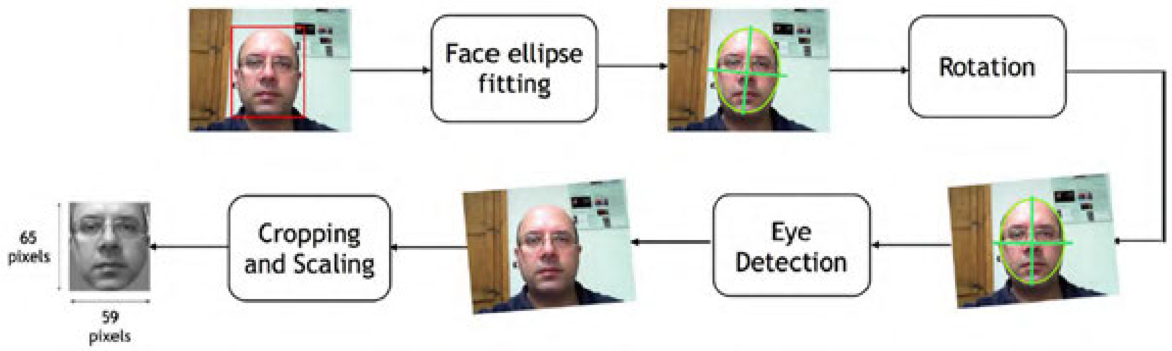


Figure 2.4: A face registration framework where the face is first rotated using a fitted ellipse, followed by cropping and scaling the face area using the eye positions [12].

2.3 Feature Extraction

Feature extraction techniques can either be frame or sequence-based, depending on whether a static image or a video sequence is used to extract facial features [24]. The frame-based approaches extract only the spatial information, such as the appearance and geometry of the face, which is then used to characterize the facial expression. In contrast, the sequence-based approach can extract spatial and temporal features to describe the evolution of expressions during a video sequence. Frame-based approaches are preferred due to their simplicity, and no assumption is made about how the expression evolves (an assumption for many sequence-based approaches). Thus, the development of FER systems based on static images has attracted much attention. Two main trends can be identified in developing FER systems in this category. The first trend is the development of FER systems based on the extraction of handcrafted features; the other consists of deep learning approaches.

Handcrafted techniques rely on domain-specific knowledge about the human face, such as facial skin changes (e.g., wrinkles) and the movement of facial components (e.g., the raising of the eyes and the opening/closing of the mouth during an expression of surprise). Within the handcrafted techniques, the geometrical features were developed to describe the shape and location of facial components [25, 26, 27, 28, 29]. Some work was also done to reconstruct action units and interpret their combinations as facial expressions [30]. Geometry-based FER systems were difficult to develop because they required the accurate labeling of action units and landmarks, which is time-consuming and tedious. During this time, appearance-based FER systems were being developed, giving rise to local descriptors for facial expression representation. Local descriptors have been used to extract texture information such as edges, corners, and spots that make up facial expressions and achieved the same or better results while being less tedious to develop than the geometric features [31].

Recently, deep neural networks have been studied for Facial Expression Recognition [32, 33, 34, 35]. Unlike traditional techniques, deep learning includes little knowledge about facial structure and appearance. Deep learning systems automatically detect and extract facial features based on neural network layers such as convolution, pooling, and fully-connect layers. This field is attracting more and more attention due to the improved recognition capacity offered by deep neural networks for static and sequence-based recognition. Mainly, CNN (Convolution Neural Networks) is the basic network for extracting more discriminative spatial features from face images, while LSTM (Long Short-Term Memory) and RNN (Recurrent Neural Networks) are suitable for characterizing the spatio-temporal features of sequences [10].

Although deep learning has improved the state-of-the-art in FER, it is still faced with several limitations. The need for large-scale datasets, massive computing power, and large amounts of memory make deep learning less desirable for embedded applications [36]. According to Ko [24], there is not a solid theory of CNN, so it is not easy to understand why and how they work. On the other hand, FER approaches based on handcrafted features are developed using face-physics-based models. Moreover, handcrafted techniques, such as local descriptors, generally have a smaller memory footprint and are computationally less complex. Therefore, it is arguable that, if appropriately crafted, conventional approaches can be more efficient and cost-effective than deep learning, especially for mobile platforms.

2.4 Local Descriptors

2.4.1 Overview

Local image descriptors are a common way of characterizing images for various computer vision tasks. One of the most popular local descriptors is the Local Binary Patterns (LBP) descriptor [37]. Due to its high discriminability and robustness to image variants such as scale and rotation, LBP has been a preferred choice for many classification problems. Although initially intended for texture classification, LBP has been applied to various problems such as face recognition, image retrieval, and Facial Expression Recognition [11]. Initially, LBP variants were preferred due to their simplicity, robustness to illumination variation, and ability to represent various facial textures. However, these approaches have been limited because they do not discriminate useful from non-useful textures, making them sensitive to noise and positional variations. LBP has inspired the development of many other local descriptors for FER, even outperforming the state-of-the-art deep learning approaches [38, 39].

Feature extraction based on local descriptors has two main components: local variation coding and local feature representation [11]. Local variation coding aims at encoding a pattern (usually a binary pattern) of image properties around a local region (usually around a pixel). Local feature representation aims at capturing the distribution of the local variation codes in a sub-region (or sub-block) expressed as a histogram. Finally, the histograms of all the sub-regions are concatenated to obtain a feature vector. Local variation coding is a key component of local descriptors, and its effectiveness in enhancing micro-patterns mostly depends on the image used for variation coding. Thus, the approaches in the literature can be differentiated based on the image properties they exploit to encode patterns. Table 2.1 summarizes the research on local descriptors designed for FER over the last two decades. Based on this research, local descriptors can be grouped into three main categories:

- Intensity-based approach
- Gradient-based approach
- Frequency, phase, and energy-based approach

2.4.2 Intensity-based Approach

The simplest form of local description compares the gray-levels or intensities of image pixels in a local region. For instance, LBP is applied by thresholding each pixel in a textured image and replacing them with binary codes. During the thresholding operation, the pixel value g_c is compared with P neighboring pixels and each Neighbor is assigned a binary value as follows: if the neighboring pixel is less than the threshold it is assigned the value 0, otherwise it is assigned the value 1. Figure 2.5 shows as example, thresholding a 3×3 local patch with 8 neighbors, producing an 8-bit code. In this case, the code has a dimension of 256. Several additions to the original LBP such as the definition of a circular neighborhood and uniform patterns have been introduced to reduce the dimension of LBP leading to a more compact representation [57]. The LBP descriptor is powerful because it can identify micro-structures such as edges, lines, spots, and flat areas. Figure 2.6 gives examples of LBP codes

Table 2.1: A summary of local descriptors for Facial Expression Recognition

#	Ref.	Year	Complete name	Abbrev.	Image Property	Dimension	Strength	Limitation
1	[40]	2011	Local Monotonic Pattern	LMP	Intensity	256	-	-
2	[41]	2014	Median Binary Pattern	MTP	Intensity	512	-	-
3	[42]	2014	Local Arc Pattern	LAP	Intensity	272	Better dimensionality, speed, and efficiency than LBP.	-
4	[43]	2012	Local Transitional Pattern	LTrP	Intensity	-	-	-
5	[44]	2008	Centralized Binary Pattern	CBP	Intensity	-	-	-
6	[45]	2014	Local Gradient Pattern	LGP	Intensity	8	Stable to monotonic illumination changes.	Susceptible to noise.
7	[46]	2016	Local Dominant Binary Pattern	LDBP	Intensity	67	-	-
8	[47]	2019	Local Prominent Directional Pattern	LPDP	Gradient magnitude and orientation	-	More resistant to positional variation and noisy edges.	-
9	[48]	2012	Gradient Directional Pattern	GDP	Gradient orientation	256	More stable and more information than intensity-based methods.	-
10	[49]	2013	Gradient-based Local Ternary Patterns	GLTeP	Gradient magnitude	512	Resistant to illumination variations and noise than intensity-based methods	Threshold setting is user-dependent.
11	[7]	2012	Local Directional Pattern Variance	LDPv	Edge responses	56	Insensitive to noise and non-monotonous illumination changes.	-
12	[50]	2012	Local Directional Number	LDN	Edge responses	56	More stable against noise and illumination variations.	-
13	[51]	2012	Local Gradient Increasing Pattern	LGIP	Edge responses	37	Stability to noise and non-monotonic illumination changes.	-
14	[52]	2013	Local Gray Code Pattern	LGCP	Edge responses	-	-	-
15	[53]	2010	Local Directional Pattern	LDP	Edge responses	56	Less sensitive to noise illumination and noise.	
16	[39]	2022	Local Direction Feature Structure	LDFS	Edge responses	-	-	-
17	[54]	2012	Binary Pattern of Phase Congruency	BPPC	Phase congruency	1062	Encodes expression specific features.	Phase congruency leads to high dimensionality and computational costs.
18	[55]	2013	Local Gabor Transitional Pattern	LGTrP	Gabor transform	256	-	High computational complexity.
19	[56]	2012	Local Gabor Directional Pattern	LGDiP	Gabor transform	280	-	High computational complexity.

obtained in a 3×3 pixel patch; 1 and 8 represent a spot or flat area, while 2, 3, 4 and 5 represent edges of varying degrees of curvature.

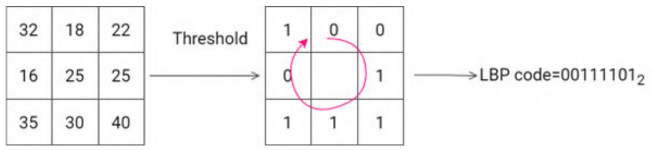


Figure 2.5: An example of LBP computation on a 3×3 neighborhood. The threshold is 25 and the labels are read clockwise to produce a binary pattern of 00111101.

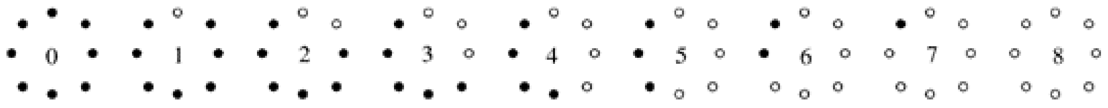


Figure 2.6: Examples of LBP codes that occur in a circular symmetric neighborhood. Black and white circles correspond to bit values of 0 and 1 respectively [57].

LBP can be formally defined as follows. Let I be a texture image, x_c be the gray-level intensity of the center pixel and x_j be the neighboring pixels in local patch of P pixels. The LBP function is defined by Equation (2.1).

$$LBP = \sum_{j=0}^{P-1} S(x_j - x_c)2^j \tag{2.1}$$

where $S(A)$ is defined as

$$S(A) = \begin{cases} 1 & \text{if } A \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

The Median Ternary Pattern (MTP) combines the advantages of median filter and the quantization of gray-scale values into three levels [41]. The code is more stable in the presence of noise and illumination variation than Local Binary Patterns and Median Binary Patterns. However, MTP depends on a user-specified threshold. Mohammad and Ali [40] proposed the Local Monotonic Pattern (LMP) to find the monotonic intensity transitions of neighboring pixels at different radii and encodes them similarly to LBP. Islam [45] presented the Local Gradient Pattern (LGP) based on the differences in gray-levels of opposite pixels across the center of a 3×3 region. The pixel differences can be seen as the local gradient flow across the center pixel. The local gradient flow is encoded as two different binary patterns concatenated to form a 4-bit code. The main contribution of this approach is its simplicity and tiny feature vector length. In [43], the Local Transitional Pattern (LTrP) is proposed. It compares the transition of intensity change at different levels of neighboring pixels. Santra and Mukherjee [46] proposed a new descriptor called the Local Dominant Binary Pattern (LDBP) for Facial Expression Recognition of faces in multiple views. LDBP encodes an LBP-like pattern in a local neighborhood but only in dominant directions of a pixel.

2.4.3 Gradient-based Approach

Gradient-based approaches examine the gradient/edge information in an image, enabling them to extract more expression-specific features than the intensity-based approaches. In this approach, edge responses are computed by applying image filters. For example, the Local Directional Pattern (LDP) descriptor uses eight Kirsch compass masks to compute the edge responses in 8 directions and encodes their relative strengths [53]. LDP assigns an 8-bit binary code to each pixel in an image using a thresholding operation like the LBP operator. For each pixel, eight directional edges responses $\{m_i\}_{i=0}^{7}$ are computed by applying Kirsch masks (M_i) in eight different orientations centered on the pixel's position. Figure 2.7 shows the eight Kirsch edge masks in eight directions. Then the top-k directional responses are assigned a 1 while the remaining $8 - k$ responses are assigned a 0. The selection of k prominent directional responses makes the binary code resistant noise and non-monotonic illumination changes. LDP is formally defined by Equation (2.2).

$$LDP_k = \sum_{i=0}^{7} B(m_j - m_k)2^j \tag{2.2}$$

where $B(a)$ is defined as

$$B(a) = \begin{cases} 1 & \text{if } a \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Similarly, the Local Direction Number (LDN) extracts directional information using the edge responses in 8 directions and encodes it using the prominent direction indices and signs [50]. The direction information was computed using two types of compass masks (derivative-Gaussian and Kirsch

$$\begin{array}{cccc}
\begin{bmatrix}-3 & -3 & 5\\ -3 & 0 & 5\\ -3 & -3 & 5\end{bmatrix} & \begin{bmatrix}-3 & 5 & 5\\ -3 & 0 & 5\\ -3 & -3 & -3\end{bmatrix} & \begin{bmatrix}5 & 5 & 5\\ -3 & 0 & -3\\ -3 & -3 & -3\end{bmatrix} & \begin{bmatrix}5 & 5 & -3\\ 5 & 0 & -3\\ -3 & -3 & -3\end{bmatrix}\\
\text{East, } M_0 & \text{North-East, } M_1 & \text{North, } M_2 & \text{North-West, } M_3\\
\begin{bmatrix}5 & -3 & -3\\ 5 & 0 & -3\\ 5 & -3 & -3\end{bmatrix} & \begin{bmatrix}-3 & -3 & -3\\ 5 & 0 & -3\\ 5 & 5 & -3\end{bmatrix} & \begin{bmatrix}-3 & -3 & -3\\ -3 & 0 & -3\\ 5 & 5 & 5\end{bmatrix} & \begin{bmatrix}-3 & -3 & -3\\ -3 & 0 & 5\\ -3 & 5 & 5\end{bmatrix}\\
\text{West, } M_4 & \text{South-West, } M_5 & \text{South, } M_6 & \text{South-East, } M_7
\end{array}$$

Figure 2.7: Kirsch edge masks in eight orientations [50].

masks). The study found that Kirsch masks produce a more suitable code for expression recognition. Furthermore, the LDN code is more stable to illumination changes and noise than the Local Directional Pattern and Local Derivative Pattern.

The Local Gradient Increasing Pattern (LGIP) defines a gradient image by replacing each pixel with an 8-bit code according to the signs of eight responses computed from Sobel masks [51]. The Sobel gradient operator produces a stable code faster than the Kirsch edges operator because the former requires less computations. The Gradient Direction Pattern (GDP) quantizes the gradient directional angles of pixels in a local neighborhood [48]. Gradients are computed by convolving the image with two Sobel masks. GDP features are more stable and retain more information than gray-level-based methods. However, the quantization depends on a user-specific threshold. Ahmed and Hossain [49] developed the Gradient-based Local Ternary Pattern (GLTeP) by calculating the gradient magnitudes of the local neighborhood using two Sobel masks and quantizing those values in three discrimination levels. The method is robust to illumination variations but depends on a user-set threshold. Other masks have been exploited. For example, Islam et al. [52] proposed the Local Gray Code Pattern (LGCP) that finds the edge responses in eight directions using the Robinson compass mask and generates a code from the relative strength of the responses. Recently, Vidyarani and Math [39] proposed the Local Direction Feature Structure (LDFS) that generates codes using the edge responses in four directions and their matching feature structure. The specifics of the edge detection algorithm are not given.

2.4.4 Frequency, Phase and Energy-based Approach

An image's frequency, energy, and phase content has also been investigated. For example, the Binary Pattern of Phase Congruency (BPPC) descriptor applies the LBP descriptor to oriented Phase

Congruency (PC) images [54]. PC images use the local phase information to enhance certain feature types such as ridge, valley, and step edges, which are later encoded by the LBP descriptor. However, the phase congruency calculation leads to high dimensionality. The Local Gabor Directional Pattern (LGDP) combines both frequency and gradient properties by applying the Gabor filters over an image, following convolutions to obtain the edge responses at each pixel. The code is generated in an LDP fashion [56]. The drawback of the Gabor wavelets is a greater time and memory requirement. Ashan et al. [55] combined an image's Gabor wavelet transform and the local transitional codes to create the Local Gabor Transitional Pattern (LGTrP). The method suffers from high computational cost due to the computation of Gabor responses in eight directions.

BPPC can be more robust against illumination and contrast variations than the gradient-based approaches due to its ability to describe various image features at different scales and orientations. However, using multiple orientations in the PC calculation increases the computational costs.

To calculate the PC of a signal, its magnitude and phase information at a particular frequency must be preserved. This can be achieved using two linear phase quadrature filters, such as the logarithmic Gabor wavelet [58]. If I denotes the signal and M_n^e and M_n^o represent the even-symmetric and odd-symmetric wavelet at a scale n, respectively, then responses of each quadrature pair of filters are expressed by the following vector:

$$[e_n(x), o_n(x)] = [I(x) * M_n^e, I(x) * M_n^o]$$

As a result, the amplitude A_n and phase ϕ_n information of the transform are defined by Equations (2.3) and (2.4).

$$A_n(x) = \sqrt{e_n(x)^2 + o_n(x)^2} \tag{2.3}$$

and

$$\phi_n(x) = atan2(e_n(x), o_n(x)) \tag{2.4}$$

respectively.

Then PC is defined as the local energy normalized by the sum of Fourier amplitude components. PC is defined by Equation (2.5).

$$PC(x) = \frac{E(x)}{\sum_n A_n} \tag{2.5}$$

where $E(x)$ is defined as

$$E(x) = \sqrt{F(x)^2 + H(x)^2}$$

with

$$F(x) = \sum_n e_n(x) \text{ and } H(x) = \sum_n o_n(x).$$

The expression of PC can be extended to account for noise and division by zero as follows:

$$PC(x) = \frac{\lfloor E(x) - T \rfloor}{\sum_n A_n(x) + \epsilon} \tag{2.6}$$

where the symbol $\lfloor \cdot \rfloor$ denotes that the enclosed quantity is not allowed to be negative, T is the threshold and ϵ is a small constant.

2.5 Classification

2.5.1 Overview

Machine learning is concerned with creating a dataset by collecting examples of some phenomenon, and then building a model based on that dataset. Classification is the use of machine learning in determining to which of a set of classes an unidentified object belongs [59]. A model for performing such as task is called a classifier. The ideal classifier must group data into categories so that data of the same class are as similar as possible, and the classes are as distinct as possible[1].

Several learning strategies exist for building a classifier, including supervised and unsupervised learning. The main difference is that supervised learning uses labeled examples, while unsupervised learning uses unlabeled examples. In supervised learning, the dataset is a collection of labeled examples $\{x_i, y\}_{i=0}^{N}$, where x is the feature vector and y is the label. Supervised learning aims to produce a model that takes a feature vector x as input and outputs a value for labeling that feature vector. Different classifiers differ based on their structure and mode of operation. For example, the Support Vector Machine finds a separating hyperplane that separates the feature space into two regions. In contrast, the k-Nearest Neighbor finds the best values of a parameter k to use in a voting system that decides which class an observation belongs to.

[1]In this section, the terms ”example”, ”feature vector” and ”data point” are used interchangeably.

By contrast, unsupervised learning uses datasets of unlabeled examples $\{x_i\}_{i=0}^{N}$ to create a model that outputs a variable or another vector that can solve a particular problem. Clustering is a type of unsupervised learning where the model groups data points into clusters based on their similarities and assigns ids to each cluster. The main challenge of the clustering approach is deciding the optimal number of clusters since labels are not provided during the training stage. Hence, deciding whether a model is optimal is more complicated in clustering than in supervised learning [60]. In centroid-based algorithms such as k-means, the goal is to find k feature vectors, called centroids, that divide the feature space into clusters. Each sample in the dataset is assigned a cluster based on its distance from the centroid of that cluster.

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is an algorithm that groups data points based on their density, i.e., how closely packed they are [61]. DBSCAN defines two hyper-parameters: ϵ and n. For a given point x, ϵ is the maximum distance between x and its neighbors, and n is the minimum number of neighbors required to place them in the same cluster as x. Essentially, ϵ determines the density required to build a cluster. The main drawback of DBSCAN is the fixed nature of ϵ, which does not deal well with clusters with varying densities. Other clustering forms include spectral and hierarchical clustering [60].

2.5.2 Facial Expression Classification

The last stage of the FER pipeline is classification, aiming to classify feature vectors as a prototypic emotional expressions. Many existing approaches classify the basic emotions such as fear, anger, surprise, happiness, disgust, and sadness [62, 63]. However, the set of emotions varies from one application to the other. In the real world, the number and type of emotions vary from one application to the other. In [64], a facial expression monitoring system predicts a patient's sudden movement during radiotherapy. Gaussian mixture model and a support vector machine are employed to classify eight basic expressions (e.g., happiness and anger). The eight expressions are further classified as uncomfortable and comfortable based on a rule. Dang et al. [65] studied the relation between facial expression and customer impression of service quality by classifying seven basic expressions using support vector machines (SVM). In [66], an entertainment robot system that can interact with humans is proposed. The RBF (Radial Basis Function) network is proposed to solve a three-class problem with positive, negative, and neutral expressions.

Several classification techniques have been applied to FER. According to Revina and Emmanuel [67], the major categories include the distance-based methods (dLHD, Euclidean Distance, MDC, KNN), Support Vector Machine, statistical methods (HMM, HCRF), rule-based methods (Decision Tree, CART), clustering based (LVQ), neural networks (MFFNN, BNN) and deep learning (CNN, DBN) methods.

K-Nearest Neighbors

K-Nearest Neighbors (KNN) method classifies a new data point by giving it the same class label as the most frequent label among the neighbors closest to it in the feature space. A constant, k, defines the number of neighbors considered to classify a new feature vector. Figure 2.8 illustrates the KNN with two classes (red and blue), and k is set to 5. The new sample is represented by the black dot and surrounded by three blue and two red dots. Hence, it will be assigned to the blue class.

The KNN classifier has the advantage of zero training costs and good performance on both small and large datasets. However, it becomes computationally expensive to find the k neighbors when classifying high-dimensionality features.

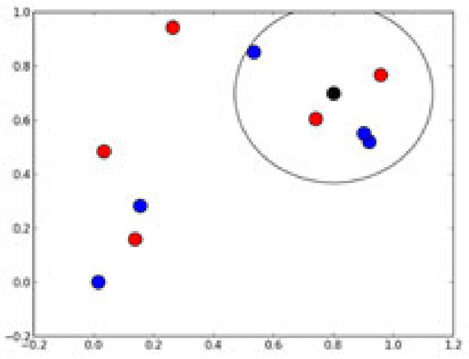


Figure 2.8: KNN example with two classes (red and blue) and $k = 5$ [68].

Support Vector Machines

Support Vector Machines (SVM) have become the standard for many FER approaches due to their advantages, such as fast training and no direct probability estimation [67, 69, 70]. An SVM is a machine learning algorithm that classifies input features by defining a separating hyperplane. Figure 2.9 illustrates how a straight line could classify data into two classes. Given l training data vector

$x_i = (x_1, x_2, \ldots, x_l)$ and the corresponding labels $y_i \in \{-1, 1\}$ the following primal optimization problem is defined as

$$\min_{w,b,\xi} \Big\{ 1/2 w^\top w + C \sum_{i=1}^{l} \xi_i \Big\} \tag{2.7}$$

subject to

$$y_i(w^\top \phi(x_i) + b) \geq 1 + \xi_i$$
$$\xi_i > 0, \quad i = 1, \ldots, l$$

where ξ_i represents the misclassification error for the i-th training vector, ξ is the total misclassification error; w is the normal vector to the hyperplane; $\frac{b}{\|w\|}$ represents the offset of the hyperplane from the origin along the normal vector w ($\|\cdot\|$ being the norm operator); $\phi(x_i)$ is the kernel function, mapping x_i into a higher dimensional space and C is the regularization parameter.

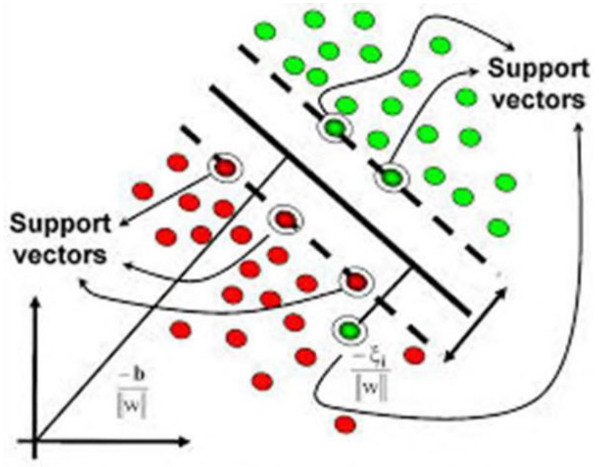


Figure 2.9: Binary classification using SVM [69].

Naïve Bayes Classifier

The Naïve Bayes (NB) classifier assigns a label $y \in \{0, 1, \ldots, M\}$ to a feature vector $X \in R^n$ based on the maximum likelihood framework defined by Equation (2.8).

$$\hat{y} = \underset{y}{argmax} P(X|y) \tag{2.8}$$

By assuming that the features in X are mutually independent given a class label y, Equation (2.9) can be used instead.

$$\hat{y} = \underset{y}{argmax} \prod_{i=1}^{N} P(x_i|y) \tag{2.9}$$

Naive Bayes classifiers differ based on how they model the probability distribution of features given a class label $(P(x_i|y))$. The Gaussian Naïve Bayes classifier is the most common and assumes a Gaussian distribution which is defined by Equation (2.10). The Cauchy distribution [71] has also been investigated for FER.

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right) \tag{2.10}$$

where x_i is the i-th feature, and σ_y and μ_y are the standard deviation and mean of the Gaussian distribution of the target label y, respectively.

Multi-Layer Perceptron

Multi-Layer Perceptron (MLP) is a basic neural network that learns a non-linear model to map training features $X(x_1, x_2, \ldots, x_m)$ to output targets y. The model comprises one input layer, one or more hidden layers, and one output layer. Each layer comprises neurons that transform the previous layer's value with a weighted linear summation $w_1x_1 + w_2x_2 + \ldots + w_mx_m$ followed by a non-linear activation function $g(\cdot) : R \rightarrow R$. Figure 2.10 shows an MLP with one hidden layer.

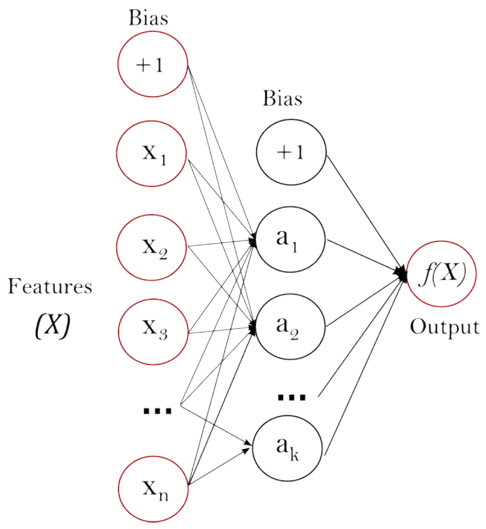


Figure 2.10: MLP neural network with one hidden layer [68].

Factors Affecting Classification

Different classification techniques have different strengths and weaknesses, making them more useful in certain contexts than others. Neural networks, for example, have a strong clustering ability, while SVM works well with high dimensionality data, making them ideal for classifying local descriptor features [72]. Hidden Markov Models (HMM) can easily model time dependencies and tackle emotion recognition in image sequences [73]. Choosing a classification method should depend on factors such as the type and dimensionality of features, the number of classes, and the application.

2.6 Model Evaluation

2.6.1 Overview

Evaluating the performance of machine learning solutions is essential [74]. Some evaluation methods may be more suitable for one problem than others. Therefore, it is essential to know the pros and cons of different methods through performance evaluation. Furthermore, a model may give satisfactory results when using one metric but may give poor results when evaluated against others. Thus, it is also essential to identify the most appropriate metrics for evaluating each machine learning problem.

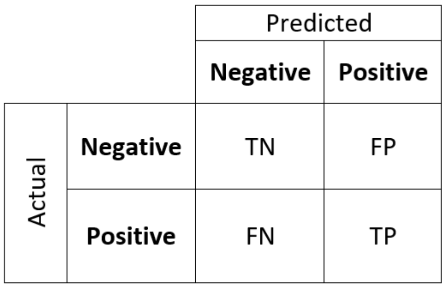
FER systems applied in some areas of life, such as medicine or surveillance, must be highly accurate, and any mistake may be life-threatening. On the other hand, applications such as online learning and video gaming are not as severely impacted by classification mistakes. Therefore, one must also consider a model's application context when judging between an acceptable and unacceptable level of performance.

2.6.2 Evaluation Metrics

Empirical studies show that it is difficult to decide which metric to use for different problems [75]. The accuracy metric is a basic performance measure in machine learning algorithms but is inappropriate in case of imbalanced data, and error costs may vary depending on the application [76]. This section reviews the most common metrics used in evaluating FER systems.

Confusion Matrix

The confusion matrix is probably the most popular way of describing a model's performance. It reveals the dependencies between the actual classes of samples and those predicted by a model [77]. Consider the confusion matrix of a binary classification problem as shown in Figure 2.11. Let the two classes be named Positives and Negatives, and the columns represent the predicted classes for each class, while the rows represent the actual classes. Four important terms can be defined: True Positives, False Positives, True Negatives, and False Negatives. True Negatives (TN) are the cases that are correctly predicted as Negative, True Positives (TP) are the cases that are correctly predicted as Positives, and False Positives (FP) are the cases that are incorrectly predicted as Positive. The False Negatives (FN) are the cases that are incorrectly predicted as Negative. According to Danjuma [78], performance evaluation often involves a trade-off between True Positives and True Negatives, and between recall and precision.



		Predicted	
		Negative	**Positive**
Actual	**Negative**	TN	FP
	Positive	FN	TP

Figure 2.11: Confusion matrix.

The confusion matrix allows the definition of other important statistical measures, such as accuracy, recall, precision, and f1-score, defined as follows:

$$\textit{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.11}$$

$$\textit{Precision} = \frac{TP}{TP + FP} \tag{2.12}$$

$$\textit{Recall} = \frac{TP}{TP + FN} \tag{2.13}$$

and

$$\textit{F1-score} = \frac{2 * \text{precision} * \text{recall}}{\text{precision + recall}}. \tag{2.14}$$

Receiver Operating Characteristics

Receiver Operating Characteristics (ROC) curve summarizes a classifier's performance over a range of trade-offs between true positive and false positive error rates. The Area Under the Curve (AUC) is the metric for the ROC curve. The larger the area, the better the model's performance. Figure 2.12 gives an example of a ROC curve.

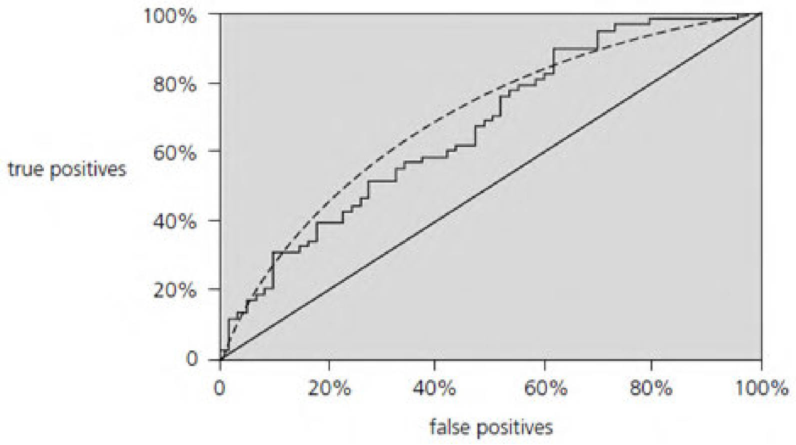


Figure 2.12: Example of a ROC curve [79].

2.6.3 Evaluation Schemes/Strategies

Train-Test Split

The most basic method for conducting a supervised learning task is dividing the available data into training and testing sets. The training set is fitted to the model to learn how to make predictions. Then, the model is used to make predictions on the test data, and the performance is evaluated by computing metrics such as accuracy and recall. This strategy can be appropriate for large enough datasets because the model is trained only once. On the other hand, this method is less desirable for small datasets because it reduces the amount of data available for the training stage, which reduces the model's learning potential. The train-test split can be done in various ways, e.g., 80% training and 20% testing, 70% training and 30% testing, etc. The 50- 50% train-test split is one of the most challenging because the amount of training data is significantly reduced [15].

Cross-validation

Cross-validation (CV) is a statistical method used to evaluate model performance. In this evaluation scheme, the data is divided into multiple subsets called folds, and each fold is evaluated once. The most common variants include k-fold [80] and Leave-One-Subject-Out (LOSO) [11]. In k-fold cross-validation, the data is first divided into k subsets (known as the "folds") containing an equal number of samples. Following this, k identical models are trained in successive runs. In each run, the model is fitted on $k-1$ of the folds, and the remaining fold is used for testing. To illustrate how k-fold works, consider an example where $k = 5$. Figure 2.13 gives training and testing folds for each run. In the first run, the model is trained on folds 1, 2, 3, 4 and tested on fold 5; In the second run, the model is fitted on folds 1, 2, 3, 5 and fitted on fold 4; In the third run, the model is fitted on folds 1, 2, 4, 5 and fitted on fold 3; and so on. After every run, a confusion matrix is computed by finding true positives, true negatives, false positives, and false negatives. Finally, the average of all confusion matrices is calculated. LOSO is a variant of cross-validation where the dataset is divided into k subsets, and each subset contains a single subject. Therefore, every subject gets tested once. The model's accuracy is estimated by averaging the recognition rate over all the subsets or folds [74].

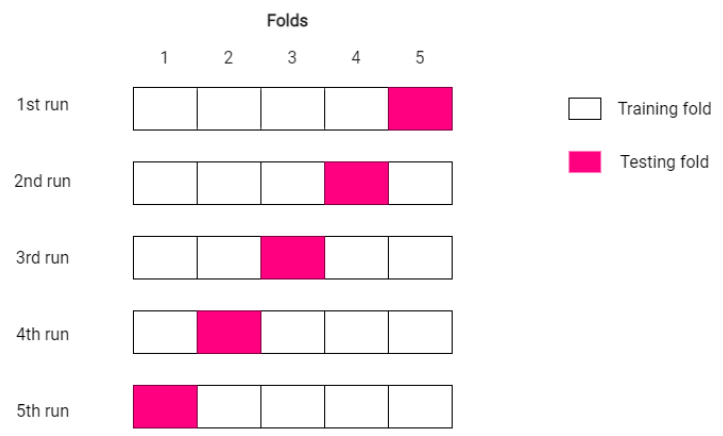


Figure 2.13: An example of a 5-fold CV showing how data is divided into training and testing folds in each run.

2.7 Databases

2.7.1 Cohn Kanade

The Cohn Kanade (CK) dataset is one of the oldest and most popular databases for benchmarking facial expression analysis systems [81]. The authors identified several challenges that every FER system faces. These include the description level, transitions among expressions, eliciting conditions, reliability and validity of training and test data, individual differences in subjects, head orientations, scene complexity image characteristics, and relation to non-verbal behavior.

2.7.2 Extended Cohn Kanade

The Extended Cohn Kanade (CK+) dataset is the extended version of the CK database, containing 593 image sequences from 123 subjects [82]. The database contains posed and non-posed images taken under lab-controlled conditions. The experiments carried out in this research only used the posed images where the subjects performed six expressions.

2.7.3 Database of Facial Affect from the InterNet

The Database of Facial Affect from the InterNet (AffectNet) is the largest facial expression, valence, and arousal database in the wild, with over a million face images manually annotated for the presence of seven discrete facial expressions [83].

2.7.4 Bahcesehir University Multimodal Affective Database

The Bahcesehir University Multimodal Affective Database (BAUM-2) dataset is one of the world's most challenging databases for evaluating facial expression systems [84]. The database contains face images that simulate real-world conditions such as multiple head poses, illumination conditions, accessories, temporary occlusions, and subjects with a wide range of eyes.

2.7.5 Japanese Female Facial Expression

The Japanese Female Facial Expression (JAFFE) dataset has 213 images of 7 facial expressions posed by 10 Japanese female models [85].

2.7.6 Multimedia Understanding Group

The Multimedia Understanding Group (MUG) dataset was introduced for the development and the statistical evaluation of Facial Expression Recognition systems [86]. The database contains face images of 80 subjects performing both posed and induced expressions with high resolution and no occlusions. There are also 80 facial landmarks points for a significant number of frames. The subjects performed the six basic expressions as defined in the FACS manual.

2.7.7 UNBC-McMaster Shoulder Pain Expression Archive Database

The UNBC-McMaster Shoulder Pain Expression Archive database was formed by capturing 200 video sequences of participants suffering from shoulder pain [87]. The database contains 48398 FACS-coded frames, associated pain scores on a frame and sequence level, self-reported and observed, and 66-point AAM landmarks. This is one of the most used databases for building FER systems in the context of pain assessment.

2.7.8 RadBoud Faces Database

The Radboud Faces Database (RFD) dataset is a popular and publicly available facial expression set containing Caucasian adult and children's images [88]. This database contains face images of various expressions, gaze directions, and head orientations in a lab-controlled environment.

2.8 Chapter Summary

Research in Facial Expression Recognition (FER) has benefited from developing local descriptors due to their ability to represent textures (e.g., edges and corners), which make up facial components active during emotion elicitation. With the above review, the following conclusions can be drawn:

- For effective Facial Expression Recognition, it is essential to extract the expression-specific features while discarding the meaningless ones. Failure to do this leads to descriptors with high dimensionality, computation cost, and high sensitivity to noise and occlusions.
- Edge-based local descriptors methods can achieve the best balance between discriminability, recognition accuracy, extraction speed, and robustness to image variants.
- Traditional machine learning classifiers such as SVM are competitive with the current deep learning methods as candidates for better FER detection and recognition.

CHAPTER 3
A COMPREHENSIVE STUDY OF FACIAL EXPRESSION RECOGNITION USING HISTOGRAMS OF ORIENTED GRADIENTS

3.1 Introduction

Facial Expression Recognition (FER) approaches can be distinguished based on their feature extraction method employed. The two main categories of feature extraction are the component-based approaches and the global approaches. The component-based approaches extract some facial components (e.g., nose, eyes, and mouth) or the geometric configurations of such components (or facial points), while the global approaches extract features representing the appearance of an entire face region. Though both categories are extensively investigated, both approaches have some unresolved issues. Component-based approaches are challenged by the misalignment of components in extreme expressions and high computational requirements [11]. Secondly, finding a set of descriptors that robustly characterizes facial expression traits is still challenging.

Some notable examples among the component-based approaches include Loconsole et al.'s [26] use of linear and eccentricity geometrical features to train a machine learning classifier to recognize eight emotions. Poursaberi et al. [30] proposed using Gauss-Laguerre wavelets to extract texture information from facial expressions and geometric positions of fiducial points to provide information for upper and lower facial action units. Both geometric and textural features are used in expression classification. Happy and Routray [89] developed a framework that uses appearance-based features of selected patches depending on the position of facial landmarks that are active during emotion elicitation. The global approaches, such as Zhao and Zhang's [90] use of the LBP descriptor, kernel discriminant isometric mapping and the Nearest neighbor classifier, are also abundant. In [50], the Local Directional Number (LDN) combined with SVM for expression classification is proposed. Uçar et al. [28] presented a novel algorithm that uses curvelet transform to generate a feature set and online sequential extreme learning machine for classification.

The purpose of this study is to revisit the use of the Histogram of Oriented Gradients (HOG) descriptor and Support Vector Machines (SVM) in the FER problem work done by Carcagni et al. [12]. The authors highlight that an optimal set of HOG parameters can be found that gives the best performance for FER. It inspires this study to propose a novel automatic face registration algorithm for gray and color images.

3.2 Datasets

Two publicly available face datasets are used in this study: the Extended Cohn Kanade (CK+) dataset [82] and the Radboud Faces Database (RFD) [88]. CK+ is one of the most used datasets to evaluate FER solutions because it contains subjects from a variety of ethnicities, ages, and genders. This dataset consists of 593 image sequences from 123 subjects. The dataset contains posed and non-posed images taken under lab-controlled conditions. The experiments carried out in this study only use the posed images of subjects performing six expressions.

To obtain a balanced dataset from the available sequences, the following images are selected: the last images in the sequences for anger, disgust, and happiness; the last and fourth images for the first 68 sequences related to surprise; the last and fourth from last images for the images of fear and sadness. A second dataset is created by adding images for the expression related to neutral. The number of images collected amounted to 347 and 407 for the first and the second datasets respectively.

The RFD dataset contains pictures of 67 subjects (adults and children) displaying eight emotions. One subset was created by selecting 469 images labeled with seven expressions: anger, contempt, disgust, fear, happiness, sadness, and surprise. Another subset is formed by adding 67 images labeled as neutral to the previous subset, bringing the number of images to 536 and the number of expressions to 8. Figures 3.1 and 3.2 give some images from the CK+ and RFD datasets respectively. Table 3.1 gives the organization of the datasets.

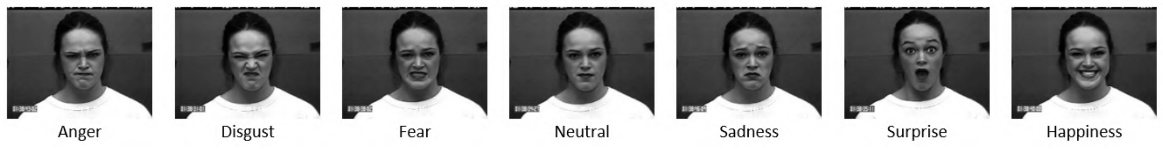


Figure 3.1: Examples of images from the CK+ dataset [82].

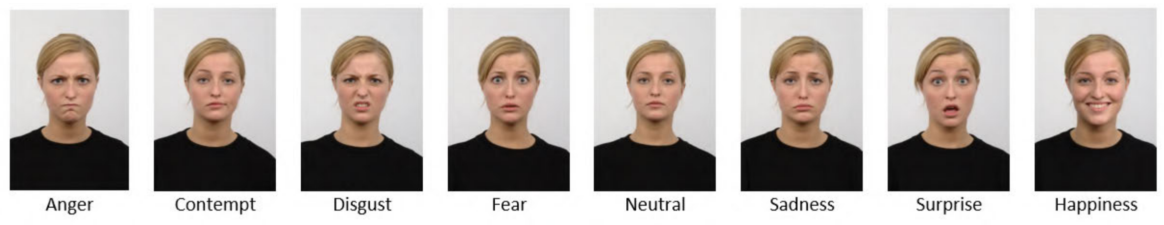


Figure 3.2: Examples of images from the RFD dataset [88].

Table 3.1: Summary of datasets

Dataset name	**Number of images**	**Number of classes**
CK+6	347	6
CK+7	407	7
RFD7	469	7
RFD8	536	8

3.3 Facial Expression Recognition Approach Overview

Several operations must be performed to recognize facial expressions from an input image. The approach adopted in this study has three stages: face acquisition, HOG feature extraction, and SVM classification. The first stage is face acquisition which involves face detection and face registration. Face detection involves locating a face region and returning its position. Face registration is performed to give the image a predefined size and ensure the eyes have a consistent position relative to the image. In the second stage, a feature vector must be extracted by applying the HOG descriptor to a registered image. Finally, the feature vector is input to a set of trained SVM classifiers to predict a facial expression in the third stage. To handle video input, a temporal window holds a fixed number of expressions as they are detected, and a decision block returns a filtered expression based on an ad-hoc rule. Figure 3.3 summarizes the proposed approach.

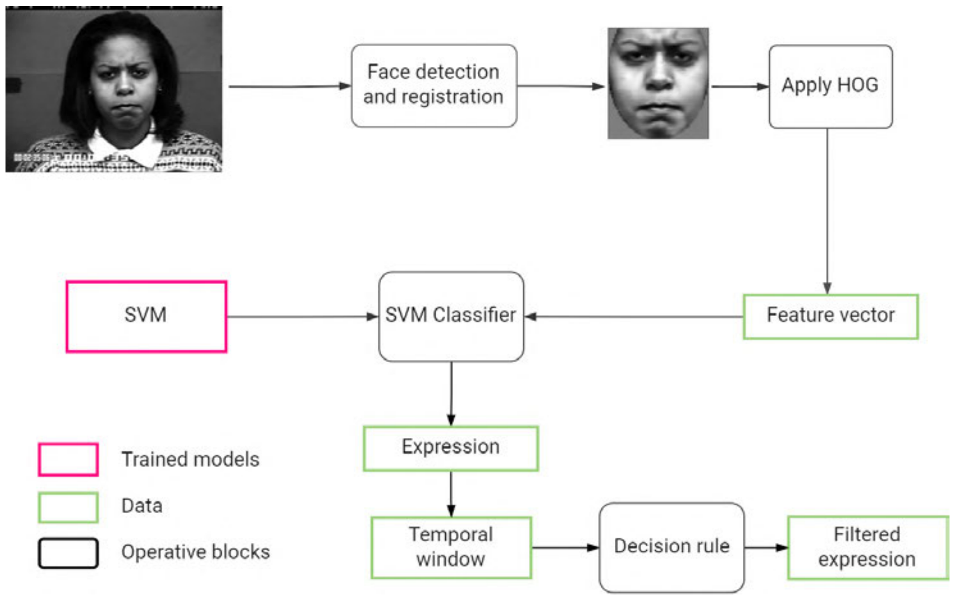


Figure 3.3: Block diagram showing the different stages of the FER approach.

3.3.1 Face Acquisition (Detection and Registration)

Face acquisition involves detecting and registering a face image. Face detection locates and returns the location of a face pattern in an input image. This is accomplished using a Viola-Jones frontal face detector. In this face detector, Haar-like features of increasingly complex constitution are extracted from an image and classified as either face or background. This study used OpenCV's [91] implementation of a Viola-Jones detector [17].

Face registration aims at giving detected faces a predefined pose, shape, and size. It is essential to register faces because the best features are usually extracted from images where the eyes are in a consistent position relative to the image. The following is a description of the face registration algorithm. First, the image is converted to gray-scale, followed by enhancement by applying contrast-limited histogram equalization. Then, a thresholding operation followed by filtration techniques produces a face blob image. Next, an ellipse is fitted around the face blob, and its angle is used to rotate the face to a vertical position. This process is not always perfect, hence, the eye locations are used to rotate the image a second time. The next step is cropping and resizing of the image. An elliptical crop followed by a rectangular crop are employed to produce an image where the eyes are in a predefined position relative to the sides of the image. Finally, the output image is resized to 59 pixels wide by 65 pixels high. Figure 3.4 and Algorithm 3.1 give the procedure for registering faces.

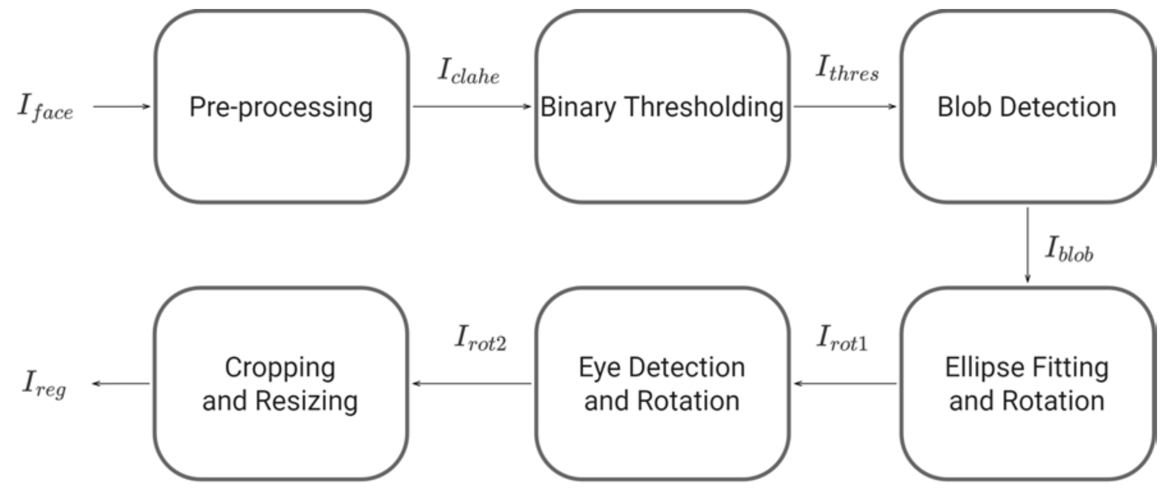


Figure 3.4: Block diagram illustrating face registration.

Algorithm 3.1 Face Detection and Registration

```
Inputs:
    $I_{face}$ : face image
Outputs:
    $I_{reg}$ : registered image
1: Pre-processing
2: Binary Thresholding
3: Blob Detection
4: Ellipse Fitting and Rotation
5: Eye Detection and Rotation
6: Cropping and Resizing
7: return $I_{reg}$
```

Pre-processing

The proposed approach for registering faces must handle gray-scale and color images. Hence, the images are first converted to the gray-scale format. Consider a gray-scale image represented by a 2D array of integers taking values between 0 (full black) and 255 (full white). Then, contrast-limited automatic histogram equalization (CLAHE) [92] is used to balance the lighting across the image. The effect CLAHE has on dark and bright images is illustrated in Figure 3.5. Let I be an $M \times N$ gray-scale image of L intensity levels, and p be the normalized histogram of I. The contrast-limited equalized image, g, is defined as

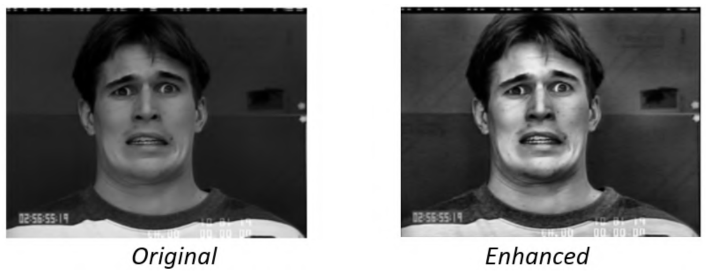
$$g_{i,j} = \left\lfloor (L-1) \sum_{n=0}^{I_{i,j}} p_n \right\rfloor \tag{3.1}$$

where p_n is defined as

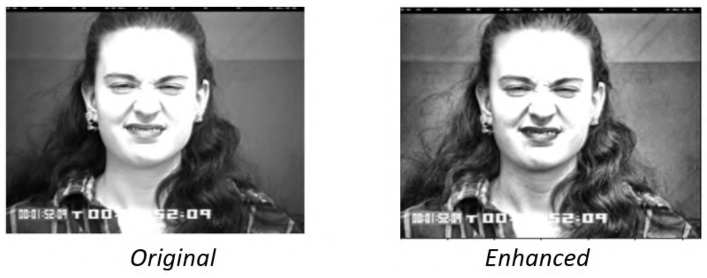
$$p_n = \frac{\sum_{i,j} A(I_{i,j} = n)}{M \times N}; \quad n = 0, 1, \ldots, L-1; \quad 0 \leq i < M; \quad 0 \leq j < N;$$

with $A(x)$ defined as

$$A(x) = \begin{cases} 1 & \text{if } x \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$



(a) Enhancing a dark image



(b) Enhancing a bright image

Figure 3.5: Effect of contrast limited automatic histogram equalization.

Binary Thresholding

Binary thresholding involves cutting out some parts of an image based on the range within which they fall. Consider an image with intensity values $I(x, y)$, then the thresholding operator B, is applied as

$$B(x,y) = \begin{cases} 1 & \text{if} \quad T_1 < I(x,y) < T_2 \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

where T_1 and T_2 are the lower and upper thresholds, respectively. These thresholds are defined as

$$T_1 = \mu - \sigma \tag{3.3}$$

and

$$T_2 = \mu + 2\sigma, \tag{3.4}$$

where μ and σ are the mean and standard deviation of the intensity values, respectively. Specifically, the mean and standard deviation are obtained by creating a histogram from the face center pattern. The mean, μ, and standard deviation, σ, are defined as

$$\mu = \frac{1}{N} \sum_{x,y} W(x,y) \times I(x,y) \tag{3.5}$$

and

$$\sigma = \frac{1}{N} \sqrt{\sum_{x,y} (I(x,y) - \mu)^2}, \tag{3.6}$$

where N is the number of pixels in an image, and $W(x, y)$ is the frequency of the bin (or interval) where the intensity $I(x, y)$ lies. Figure 3.6 shows the effect of binary thresholding and a gray-scale image.

Blob Detection

The thresholding above aims to separate facial skin pixels from the background pixels. However, this is difficult to achieve in a gray-scale image, where different objects can have similar intensities. Hence, unwanted regions must be filtered out. The first filter is an ellipse computed from the face's

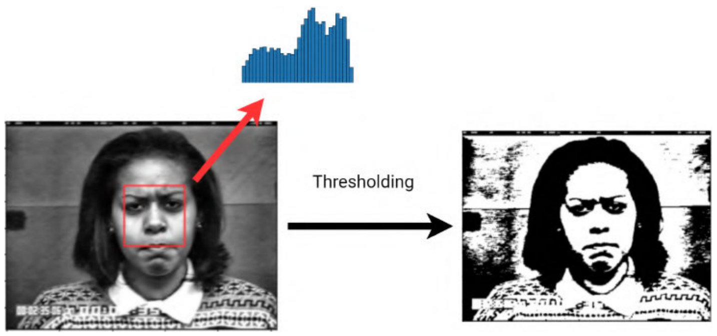


Figure 3.6: Binary thresholding.

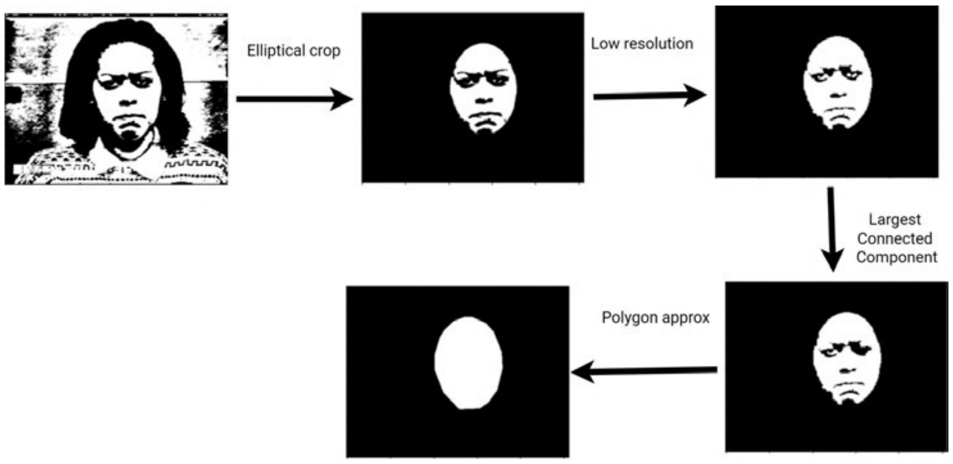


Figure 3.7: Blob detection illustrated as a series of filtering operations with the result being a filled polygon that acts a face blob.

dimensions and location. Elements inside the ellipse are retained, while elements outside the ellipse are discarded. The second filter searches for the graph-encoded binary image's largest connected components [93]. The blob's shape is then refined by estimating the polygonal shape around it. A polygon is estimated by finding a set of contour points [94] enclosing the blob, computing its convex hull [95], and approximating a polygonal shape. Polygon estimation is important because it fills holes in the blob and makes the ellipse-fitting step more reliable. The process of blob detection is illustrated in Figure 3.7.

Ellipse Fitting

A region-growing algorithm is employed to find an ellipse's parameters that fit around the face blob [91]. The ellipse's angle is used to rotate the image so that the eyes are horizontal and the face is upright and vertical. Figure 3.8 shows an ellipse fitted around the face blob, and the angle (θ) between the minor axis and the x-axis is used to rotate the image.

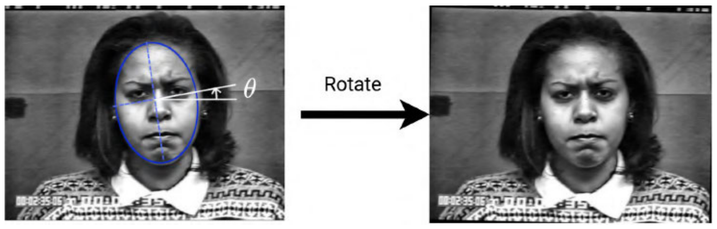


Figure 3.8: Rotate an image using the angle of an ellipse fitted around the face blob.

Eye Detection

The ellipse fitting algorithm sometimes fails to represent the face blob correctly, so the rotation angle is wrong. The image is further rotated using the eye positions to solve this problem. Once the eyes are located, the line passing through them is found, and its elevation angle is used to rotate the image as shown in Figure 3.9. If the eye detector fails to detect eyes, their positions are estimated, and no rotation is done.

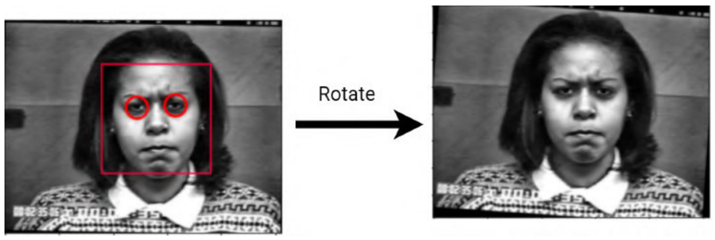


Figure 3.9: The eye locations are used to rotate the image a second time.

Cropping and Resizing

The last step of face registration involves performing an elliptical and rectangular crop around the face image and re-scaling it to a predefined size. First, the ellipse is used to crop the face region, and then the eye locations are used to provide a measure to crop a rectangular portion of the face. Figure 3.10 shows how the image is cropped and scaled using the eye locations and ellipse axes. Let (x_E, y_E) be the center of the ellipse and (x_B, y_B, w_B, h_B) represents the x-coordinate, y-coordinate, width and height of the rectangular area being cropped. The following equations can be used to calculate these parameters:

$$\Big[x_E, y_E\Big] = \Big[\frac{1}{2}(x_1 + x_2), \frac{1}{2}(y_1 + y_2) + \frac{1}{2}a\Big], \tag{3.7}$$

$$\left[x_B, y_B\right] = \left[x_E - b, y_E - 0.8a\right], \tag{3.8}$$

and

$$\left[w_B, h_B\right] = \left[2b, 1.6a\right], \tag{3.9}$$

where (x_1, y_1) and (x_2, y_2) are the eye locations, a and b are half the axes' lengths of the ellipse; a being the major axis and b being the minor axis.

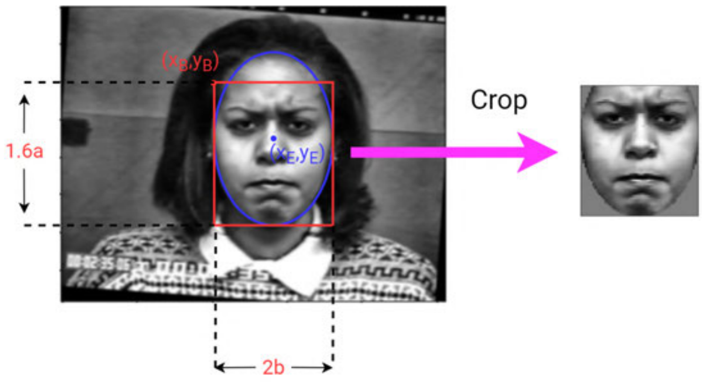


Figure 3.10: Parameters of the ellipse and rectangle used to crop and resize an image.

3.3.2 Histogram of Oriented Gradients

The Histogram of Oriented Gradients (HOG) descriptor was first proposed as a local-histogram descriptor for human detection in [96] and since then has been successfully applied to Facial Expression Recognition [97]. The HOG descriptor is calculated as follows. The first stage computes first-order image gradients by differentiating along the x and y directions. In the second stage, the first-order gradients are used to calculate the magnitude and orientation of the gradient images. In the third stage, the images are divided into cells, and a histogram is computed for each cell. Each cell histogram divides the gradient orientation range into a fixed number of intervals (or bins). The gradient magnitudes of the pixels in the cell are used to vote into the histogram (see more details in the next paragraph). In the fourth stage, the cells are grouped into larger spatially connected groups called blocks, and block descriptors are calculated. A block descriptor is created by concatenating the cell histograms in that block and normalizing them (further explained below). Note that the blocks overlap, so cells are shared between blocks and appear more than once (but with different normalizations) in the final feature vector. Finally, all the block descriptors are concatenated to form the final HOG feature vector.

Consider $L(x, y)$ as the intensity function of an $M \times N$ gray-scale image. The first-order image gradients along the x and y directions at coordinate (x, y) are defined by Equations (3.10) and (3.11).

$$I_x = L(x-1, y) - L(x+1, y) \tag{3.10}$$

$$I_y = L(x, y-1) - L(x, y+1) \tag{3.11}$$

Given the differential components, I_x and I_y, the gradient magnitudes and orientations of the image are defined by Equations (3.12) and (3.13).

$$M(x, y) = \sqrt{I_x^2 + I_y^2} \tag{3.12}$$

$$\alpha = \frac{180}{\pi}(\text{arctan2}(I_y, I_x) \bmod \pi), \tag{3.13}$$

where arctan2 is the four-quadrant inverse tangent, which yields values between $-\pi$ and π. Using the gradients magnitude and orientation, the Histogram of Oriented Gradients for each cell is defined by Equation (3.14).

$$H_{cell}(i, j, k) = \sum_{(u,v) \in C_{i,j}} M(u, v) \quad \text{if} \quad \alpha(u, v) \in \theta_k \tag{3.14}$$

$$0 < i < \frac{M}{cellSize} - 1, \quad 0 < j < \frac{N}{cellSize} - 1,$$

where $C_{i,j}$ is a set of x and y coordinates within the (i, j) cell and θ_k is the range of each bin, e.g., for 6 orientations, $\theta_1 = [165°, 180°) \cup [0°, 15°), \theta_2 = [15°, 45°), \theta_3 = [45°, 75°), \theta_4 = [75°, 105°), \theta_5 = [105°, 135°)$, and $\theta_6 = [135°, 165°)$. Therefore, $H_{cell}(i, j, k)$ represents the Histogram of Oriented Gradients at the (i, j) cell. After obtaining cell histograms, the cells are grouped into blocks and normalized as follows:

- First, the block descriptor is built by concatenating cell histograms within the block using Equation (3.15).

$$H_{block} = \left(H_{cell}^1 H_{cell}^2 \cdots H_{cell}^{n_c}\right) \tag{3.15}$$

where n_c is the number of cells per block.

- Normalize the descriptor using L2 normalization defined by Equation (3.16).

$$\hat{h_l} = \frac{h_l}{\sqrt{\sum_l h_l^2 + e^2}}, \tag{3.16}$$

where h_l is the l-th element of the block descriptor, $\hat{h_l}$ is the corresponding normalized value, and e is a constant to prevent division by zero.

- Limit the maximum value to 0.2 using by Equation (3.17):

$$h_l' = \begin{cases} 0.2 & \text{if } \hat{h_l} > 0.2 \\ \hat{h_l} & \text{otherwise} \end{cases} \tag{3.17}$$

- Repeat L2 normalization using Equation (3.16).

- Concatenate all the block descriptors to produce the final HOG feature vector as given by Equation (3.18).

$$H_{hog} = \left(H_{block}^1 H_{block}^2 \cdots H_{block}^{n_b}\right) \tag{3.18}$$

where n_b is the number of blocks.

Algorithm 3.2 summarizes the calculation of a HOG feature vector. In this study, scikit-image's implementation[1] of the HOG descriptor was used.

[1]`https://github.com/scikit-image/scikit-image/blob/main/skimage/feature/_hog.py` (accessed 31 July 2021)

Algorithm 3.2 Histogram of Oriented Gradients

```
Inputs:
    $I$ : input image
Outputs:
    $H_{hog}$ : feature vector
1: Compute first order image gradients using Equations (3.10) and (3.11).
2: Compute gradient orientations and gradient magnitudes using Equations (3.12) and (3.13)
3: Build the Histograms of Oriented Gradients for all the cells using Equation (3.14).
4: Build a descriptor for all the blocks using normalization using Equations (3.15), (3.16), and (3.17).
5: Return a feature vector by concatenating all the block descriptors using Equation (3.18).
```

3.3.3 Support Vector Machines

Once all the HOG features are extracted, a training dataset comprises feature vectors alongside the corresponding label. The datasets are used to train a classifier based on the SVM strategy. The experiments discussed here use the multi-C-support classification (SVC) strategy[2] implemented by scikit-learn's LIBSVM library.

A set of hyper-parameters must be tuned to train an SVM classifier: the kernel function, the regularization parameter (C), and the kernel coefficient (gamma). For each classifier, grid-search cross-validation is employed to find the best settings for the classifier [98]. Table 3.2 gives the candidate values for each SVM parameter. In grid-search cross-validation, all possible combinations of the parameters are used to train a classifier using 10-fold cross-validation. Then, the parameter values which produce the best average recall are selected. Table 3.3 gives the parameter values selected when training the SVM classifier with different types of feature vectors depending on the local descriptor.

[2]`https://github.com/scikit-learn/scikit-learn/blob/f3f51f9b6/sklearn/svm/_classes.py` (accessed 31 July 2021)

Table 3.2: Hyper-parameter values used for the grid-search

Hyper-parameter	**Values**
C	1, 10, 100, 1000
Gamma	5, 0.5, 0.05, 0.005
Kernel	Linear, RBF

Table 3.3: Support vector machines hyper-parameter settings for different local descriptors

Descriptor	**C**	**Gamma**	**Kernel**
HOG	1000	0.05	RBF
LBP	1000	0.05	RBF
CLBP	1000	0.05	Linear
WLD	1000	0.05	Linear

3.3.4 Temporal Analysis

A temporal window storing recent predictions and a decision block make the system suitable for video sequences. The final expression is decided by observing m expressions stored in the temporal window and checking whether at least n (provided $n \leq m$) frames are classified with the same facial expression. The expression is classified as s if the following condition is met:

$$\sum_{j=i-m+1}^{i} \Lambda(p_j, s) \geq n \tag{3.19}$$

where

$$s \in \{\textit{anger}, \textit{surprise}, \textit{happiness}, \textit{fear}, \textit{disgust}, \textit{sadness}, \textit{contempt}, \textit{neutral}\},$$

and $\Lambda(p_j, s)$ is defined as

$$\Lambda(p_j, s) = \begin{cases} 1 & \text{if } p_j = s \\ 0 & \text{otherwise} \end{cases},$$

p_j is the expression assigned to the j-th frame and i is the index of the most recent frame in the window.

3.3.5 Other Local Descriptors of Interest

The performance of the HOG descriptor was compared with three other local descriptors: Local Binary Pattern (LBP) [99], Weber's Local Descriptor (WLD) [97] and Compound Local Binary Pattern (CLBP) [100]. These local descriptors are used in many existing FER approaches.

Local Binary Pattern

Local Binary Pattern (LBP) is a local descriptor commonly used to describe textural appearance in pattern recognition problems. The LBP histogram contains information about the local micro-pattern distribution, such as flat areas and edges in an image. The following is an account of how the feature vector is computed. First, the LBP descriptor is applied to each pixel, treating it as a threshold and assigning a label based on a rule (more details in the following paragraph). This produces an LBP image, which is then subdivided into equally sized blocks, and the histogram of each block is computed. The final feature is formed by ordering and concatenating the histograms of all the blocks.

The original LBP descriptor is computed using a 3×3 grid of pixel intensities with a center pixel (called the threshold), g_c. Then a value of 1 or 0 is assigned to each of the eight surrounding pixels using the following rule: if the neighboring pixel is less than the threshold, it is assigned the value 0. Otherwise, 0 is assigned. A binary code is then read from the sequence of 0's and 1's and converted to a decimal number. The decimal number is then assigned to the center (or threshold) pixel. Once this operation has been performed on the entire image, an LBP image is obtained. The histogram of the LBP image is then used as a feature vector [37]. An example of the LBP code of a pixel in a 3×3 neighborhood block is given in Figure 2.5 under Section 2.4.2.

LBP can be extended to allow for any number of neighbors using a circular neighborhood and bi-linear interpolation of pixel values [57]. The descriptor can be extended further by restricting the LBP codes to use only uniform patterns. An LBP code is said to be a uniform pattern if it contains at most two bit-wise transitions from 1 to 0 or vice versa when considering the binary string circular e.g., 00000000 and 00011110 are considered uniform patterns. The uniform LBP operator is denoted as $LBP_{P,R}^{u2}$, where P stands for the number of neighbors and R is the circular radius and $u2$ means that only uniform patterns are retained and all remaining patterns are replaced by a single label. The LBP code at a coordinate (x, y) is given by Equation (3.20).

$$f_c(x,y) = \sum_{j=0}^{P-1} S(g_j - g_c)2^j, \tag{3.20}$$

where

$$x = 0, 1, \ldots, N-1, \;\; y = 0, 1, \ldots, M-1,$$

and g_j corresponds to the neighboring intensity values, N and M are the width and height of the input image, respectively, and $S(A)$ is defined as

$$S(A) = \begin{cases} 1 & \text{if } A \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Then, a histogram, $(H_i)_{i=0,1,\ldots,n-1}$ is defined by Equation (3.21).

$$H_i = \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} I\left(f_c(x,y) = i\right) \tag{3.21}$$

where n is the number of unique labels and $I(A)$ is defined as

$$I(A) = \begin{cases} 1 & \text{if } A \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

A more recent addition to the LBP algorithm adds spatial information to the descriptor [101]. Spatial information is encoded by subdividing the image into m equally-sized blocks, R_j. Next, the histograms of each block are concatenated into a single feature vector. The spatially enhanced histogram can be defined by Equation (3.22).

$$H_{i,j} = \sum_{(x,y)\in R_j} I\left(f_c(x,y) = i\right) \tag{3.22}$$

$$i = 0, 1, \ldots, n-1, \quad j = 0, 1, \ldots, m-1$$

Figure 3.11 and Algorithm 3.3 summarize the extraction of an LBP feature vector.

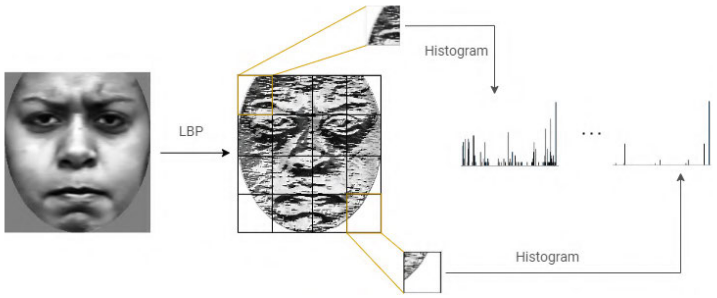


Figure 3.11: Computing the LBP features of an image.

Algorithm 3.3 Local Binary Patterns

```
Inputs:
    $I$ : input image
Outputs:
    $H = (H^0 H^1 \cdots H^{N_b - 1})$ : feature vector
1: Divide the input image into blocks, $(I_b)_{b=0,1,\ldots,N_b-1}$.
2: for each block, $b$ do
3:     Compute LBP codes, $f_c^b$, for the block $b$ using Equation 3.20.
4:     Compute a histogram, $H^b$, using Equation 3.21.
5: end for
6: Concatenate the histograms, $H = (H^b)_{b=0,1,\ldots,N_b-1}$.
7: return $H$
```

Weber's Local Descriptor

Weber's Local Descriptor (WLD) is comparable to the human perception of a pattern, which depends not only on the change of stimulus (such as sound and lighting) but also on the original intensity of the stimulus [102]. WLD consists of a differential excitation (ξ) and a gradient orientation (θ). The differential component is expressed as the ratio between two terms: the first is the relative intensity differences of a current pixel against its neighbors; the other is the intensity of the current pixel. The second component of WLD is the gradient orientation. A feature vector is obtained by rearranging the differential excitations several times into subgroups following a particular rule (further discussed below), then creating histograms of those subgroups. Finally, the histograms are reordered and concatenated into a single feature vector for classification. Figure 3.12 illustrates extracting WLD features from an input image.

Consider a 3×3 pixels block, G_s, taken from an input image, G. The WLD operator is applied as follows:

- Compute the differential excitation component, which is expressed as:

$$\xi(x_c) = \tan^{-1}\left(\frac{V_s^{00}}{V_s^{01}}\right) = \tan^{-1}\left[\sum_{i=0}^{p-1}\left(\frac{x_i - x_c}{x_c}\right)\right] \tag{3.23}$$

- The second component of WLD is the gradient orientation $\theta(x_c)$ which is defined as:

$$\theta(x_c) = \tan^{-1}\left[\frac{V_s^{11}}{V_s^{10}}\right] = \tan^{-1}\left[\frac{x_5 - x_1}{x_7 - x_3}\right] \tag{3.24}$$

provided

$$G_s = \begin{array}{|c|c|c|} \hline x_0 & x_1 & x_2 \\ \hline x_7 & c & x_3 \\ \hline x_6 & x_5 & x_4 \\ \hline \end{array} \qquad \xi(x_c) \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \qquad \theta(x_c) \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$$

- Map the gradient orientations from the interval $[-\frac{\pi}{2}, \frac{\pi}{2}]$ to the interval $[0, 2\pi]$ by applying the following function:

$$\theta' = \text{arctan2}(V_s^{11}, V_s^{10}) \tag{3.25}$$

where $\arctan2(V_s^{11}, V_s^{10})$ is defined as

$$\arctan2(V_s^{11}, V_s^{10}) = \begin{cases} \theta & \text{if } V_s^{11} > 0 \text{ and } V_s^{10} > 0 \\ \pi - \theta & \text{if } V_s^{11} > 0 \text{ and } V_s^{10} < 0 \\ \theta - \pi & \text{if } V_s^{11} < 0 \text{ and } V_s^{10} < 0 \\ 2\pi + \theta & \text{if } V_s^{11} < 0 \text{ and } V_s^{10} > 0 \end{cases}$$

$$V_s^{11} = x_5 - x_1; \quad V_s^{10} = x_7 - x_3$$

- The mapped gradients, θ', are further quantized into T dominant orientations as follows:

$$\Phi_t = f_q(\theta') = \frac{2\pi}{T}\text{mod}\left(\left\lfloor\frac{\theta'}{2\pi/T}\right\rfloor, T\right) \tag{3.26}$$

where t is a linear function that maps a differential excitation with orientation θ' to a particular dominant orientation Φ_t . Next, a 2D histogram, $(\text{WLD}(\xi_j, \Phi_t))_{j=0,1,\ldots,N-1,\ \ t=0,1,\ldots,T-1}$, is created where N is the dimensionality (number of pixels) in an image and T is the number of dominant orientations. The 2D histogram has C rows and T columns, where C is the number of cells in each orientation. Intuitively, each cell of the 2D histogram represents the frequency of a differential excitation interval on a dominant orientation.

The 2D histogram is further encoded into a 1D histogram, H as follows. First, each column of the 2D histogram is projected to form a 1D histogram $(H(t))_{t=0,1,\ldots,T-1}$. Each histogram is produced by grouping differential excitations ξ_j corresponding to a dominant orientation Φ_t.

Next, each sub-histogram, $H(t)$, is divided into M segments, $(H_{m,t})_{m=0,1,\ldots,M-1}$. The sub-histograms, $H_{m,t}$, collectively form a histogram matrix. Each row of this matrix corresponds to a differential excitation segment and each column corresponds to a dominant orientation. The histogram matrix is reorganized as a 1D histogram by first concatenating the rows of the histogram matrix to produce a histogram $H_m = \{H_{m,t}\}_{t=0,1,\ldots,T-1}$. The resulting M sub-histograms are concatenated to form a 1D histogram $H = \{H_m\}_{m=0,1,\ldots,M-1}$.

Note that each sub-histogram, $H(t)$, is divided into M segments, $H_{m,t}$, by dividing the interval of differential excitations $l = [-\frac{\pi}{2}, \frac{\pi}{2}]$ into M segments l_m. Furthermore, each sub-histogram $H_{m,t}$ is an

S-bin histogram defined as:

$$H_{m,t} = \{h_{m,t,s}\} \tag{3.27}$$

$$h_{m,t,s} = \sum_j \delta(S_j = s)$$

$$S_j = \text{mod}\left(\left\lfloor \frac{\xi_j - \eta_{m,l}}{(\eta_{m,u} - \eta_{m,l})/S} \right\rfloor, S\right), \quad \delta(A) = \begin{cases} 1 & \text{A is true} \\ 0 & \text{A is false} \end{cases},$$

$$\eta_{m,l} = \left(\frac{m}{M} - \frac{1}{2}\right)\pi \text{ and } \eta_{m,u} = \left(\frac{m+1}{M} - \frac{1}{2}\right)\pi \quad .$$

Figure 3.12 and Algorithm 3.4 summarize the computation of a WLD feature vector.

Algorithm 3.4 Weber's Local Descriptor

```
Inputs:
    $I$ : input image
Outputs:
    $H = (H^0 H^1 \cdots H^{N_b - 1})$ : feature vector
1: Divide the input image into blocks, $(I_b)_{b=0,1,\ldots,N_b-1}$.
2: for each block, $b$ do
3:     Compute the differential excitations, $\xi_j$, using Equation (3.23).
4:     Compute the gradient orientations, $\theta_j$, using Equation (3.24).
5:     Map the gradient orientation, $\theta'_j$, using Equation (3.25).
6:     Quantize the mapped orientations into $T$ dominant orientations, producing $\Phi_t$ using Equation (3.26).
7:     Create sub-histograms, $H_{m,t}$, using Equation (3.27).
8:     Concatenate the sub-histograms, $H^b = \{H_{m,t}\}_{t=0,1,\ldots T-1,\ m=0,1,\ldots M-1}$
9: end for
10: Concatenate the 1D histograms, $H = \{H^b\}_{b=0,1,\ldots,N_b-1}$
11: return $H$
```

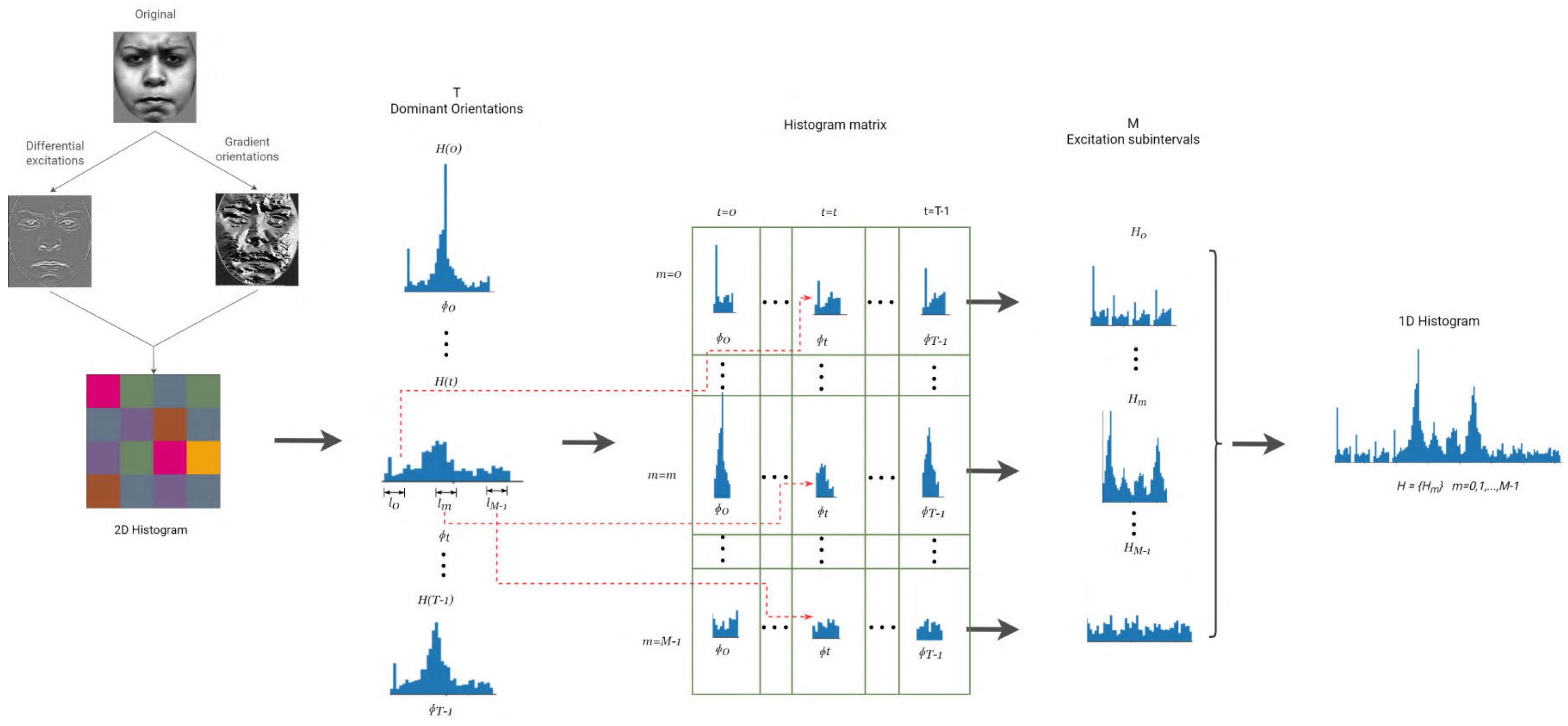


Figure 3.12: Extraction of WLD features from an input image.

Compound Local Binary Pattern

Compound Local Binary Pattern (CLBP) is an extension of the LBP descriptor. The CLBP operator differs from the original LBP in that it encodes both the magnitude and the sign of the differences between a center (or threshold) pixel and its P neighbors [100]. While LBP assigns one bit to each neighbor, CLBP uses two bits to encode the signs and magnitudes of the differences between the center pixel and its neighbors: the first bit expresses the sign of the difference in pixel values, and the second bit is for the magnitude of the difference with respect to a threshold value. Thus, compared to LBP, which produces a P-bit code, CLBP assigns a $2P$-code to the center pixel based on the intensities of P neighbors. The binary pattern is subdivided into two sub-CLBP patterns by regrouping the neighbors' binary labels (more details in the next section). A 1D histogram is created for each sub-pattern, and all are concatenated, producing a single feature vector. Figure 3.13 illustrates how the original LBP code is computed over a pixel with a circular neighborhood of 8 pixels and the resulting LBP code $(11111111)_2$. The CLBP operator is illustrated in Figure 3.14. Here, each neighbor is represented by two bits, which gives a 16-bit code when combined.

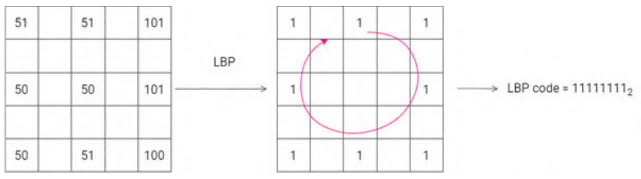


Figure 3.13: Generating a binary code using LBP, where the code is $(11111111)_2$.

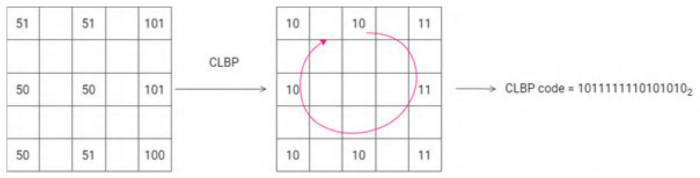


Figure 3.14: Illustration of the CLBP operator, where the code is $(1011111110101010)_2$.

The CLBP code is further split into two sub-CLBP patterns. Considering the example above, the split is performed according to the following sequences: (0, 1, 4, 5, 8, 9, 12, 13) and (2, 3, 6, 7, 10, 11, 14, 15), respectively. Figure 3.15 illustrates splitting a CLBP code into two sub-CLBP codes.

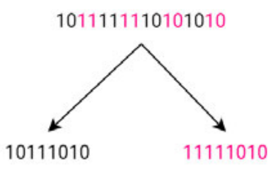


Figure 3.15: Splitting a CLBP code into two sub-CLBP codes.

In this way, the histogram obtained only encodes the occurrences of micro-patterns in an image. However, the availability of both location and spatial relationships can better characterize facial features. For this reason, the CLBP descriptor is extended by first partitioning the input image into sub-regions (or blocks) and applying the CLBP descriptor to each block. Finally, concatenating the histograms of the blocks gives the final feature vector for classification. Figure 3.16 gives the process of extracting the original CLBP feature vector.

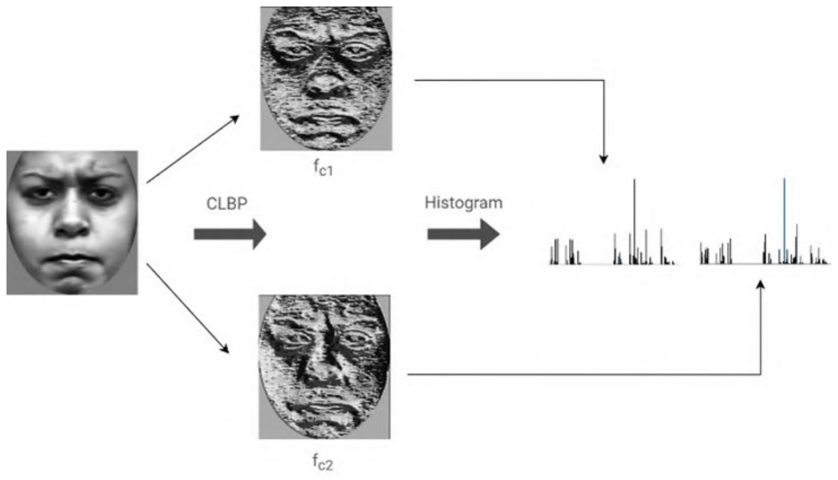


Figure 3.16: Generating a CLBP feature vector.

The following is a more detailed description of the CLBP operator. Each pixel is treated as a center having P neighbors. Each neighbor is given a 2-bit code: the first bit is set to 0 if the neighboring pixel is less than the center pixel and is set to 1 otherwise. The second bit is set to 1 if the magnitude of the difference between the neighboring pixel and the center pixel is greater than the threshold, M_{avg}. Otherwise, the bit is set to 0. The threshold value is the average magnitude of the differences between the center pixel and its P neighbors. The rest of this discussion considers $P = 8$, also used in implementing CLBP.

The average magnitudes M_{avg} can be computed as follows. First, through convolutions, apply eight filters $(K_i)_{i=0,1,...,7}$ (see Figure 3.17) to the input images, obtaining the pixel differences for the eight neighbors separately, producing eight matrices $(M_j)_{j=0,1,...,7}$. Then, compute the averages of the magnitudes of the previous matrices, obtaining M_{avg}. The matrices of pixel differences can also be used to obtain the magnitudes and signs of the differences through simple operations.

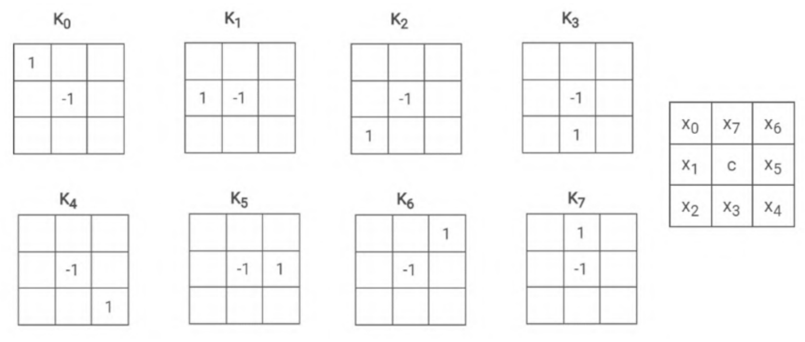


Figure 3.17: Convolution masks used to generate the CLBP feature.

Let $f_c(x, y)$ be the function representing the CLBP code of a pixel positioned at (x, y) defined as:

$$f_c(x, y) = \sum_{p=0}^{P-1} s(i_p, i_c) 2^{2p} \tag{3.28}$$

where $s(i_p, i_c)$ is defined as

$$s(i_p, i_c) = \begin{cases} 00 & \text{if } i_p - i_c < 0 \text{ and } |i_p - i_c| \le M_{avg} \\ 01 & \text{if } i_p - i_c < 0 \text{ and } |i_p - i_c| > M_{avg} \\ 10 & \text{if } i_p - i_c \ge 0 \text{ and } |i_p - i_c| \le M_{avg} \\ 11 & \text{otherwise} \end{cases}$$

i_c and i_p represent the gray values of the center c, and a neighbor p, and M_{avg} is the average magnitude of the differences between i_c, i_p pairs in the local neighborhood.

The CLBP binary code is subdivided into two sub-CLBP patterns f_{c1} and f_{c2}. Hence, during implementation, Equation (3.28) is not used but instead, Equations (3.29) and (3.30) are used. The sub-CLBP patterns are defined as follows:

$$f_{c1} = \sum_{p \in O, k \in E} \Big(S_{1p}(i_p, i_c) 2^{k+1} + S_{2p}(i_p, i_c) 2^{k} \Big) \tag{3.29}$$

$$f_{c2} = \sum_{p \in O, k \in E} \Big(S_{1p}(i_p, i_c) 2^{k+1} + S_{2p}(i_p, i_c) 2^{k} \Big) \tag{3.30}$$

$$O = \{1, 3, 5, 7\}, \;\; E = \{0, 2, 4, 6\}$$

where $S_{1p}(i_p, i_c)$ and $S_{2p}(i_p, i_c)$ are defined as

$$S_{1p}(i_p, i_c) = \begin{cases} 0 & \text{if } i_p - i_c < 0 \\ 1 & \text{if } i_p - i_c \geq 0 \end{cases}, \quad S_{2p}(i_p, i_c) = \begin{cases} 0 & \text{if } |i_p - i_c| \leq M_{avg} \\ 1 & \text{if } |i_p - i_c| > M_{avg} \end{cases}$$

with

$$M_{avg} = \frac{1}{8} \sum_{p=0}^{7} |i_p - i_c|$$

The result is two sub-CLBP patterns which are treated as separate binary codes. The binary codes are only combined in the next stage: feature vector generation. After generating the sub-CLBP patterns for each pixel, two binary codes for each pixel are obtained, thus, two encoded image representations. Next, histograms are generated from the two sub-CLBP images and concatenated them, producing a spatially combined histogram that acts as a feature vector for classification. Algorithm 3.5 summarizes the generation of CLBP feature vectors.

Algorithm 3.5 Compound Local Binary Patterns

```
Inputs:
    $I$ : input image
Outputs:
    $H = (H^0 H^1 \cdots H^{N_b - 1})$ : feature vector
1: Divide the input image into blocks, $(I_b)_{b=0,1,\cdots,N_b-1}$.
2: for each block, $b$ do
3:     Compute the sub-CLBP codes, $f_{c1}^b$ and $f_{c2}^b$, using Equations (3.29) and (3.30), respectively.
4:     Create a histogram for the codes, $H_1^b$ and $H_2^b$.
5:     Concatenate the sub-histograms, $H^b = \{H_1^b H_2^b\}$.
6: end for
7: Concatenate the 1D histograms, $H = \{H^b\}_{b=0,1,\ldots,N_b-1}$.
8: return $H$
```

3.4 Results and Analysis

3.4.1 Experimental Phase 1: Optimization of HOG Parameters

Qualitative Analysis

A qualitative analysis was done to observe the different feature outputs when the HOG parameters are varied. HOG images were generated by setting the number of orientations to 9 and varying the cell size[3] between 3, 8, and 15. Figure 3.18 gives the HOG images for a particular subject's neutral and surprised expressions. It can be seen that when the cell size is 3, though there are more cells, the edges are harder to see. When the cell size is 15, the edge information is too condensed to capture directions adequately. The best result is seen when a cell size of 8 is used since the edge information is enough to differentiate between the emotions.

[3]These parameter values are arbitrary and are used for illustration.

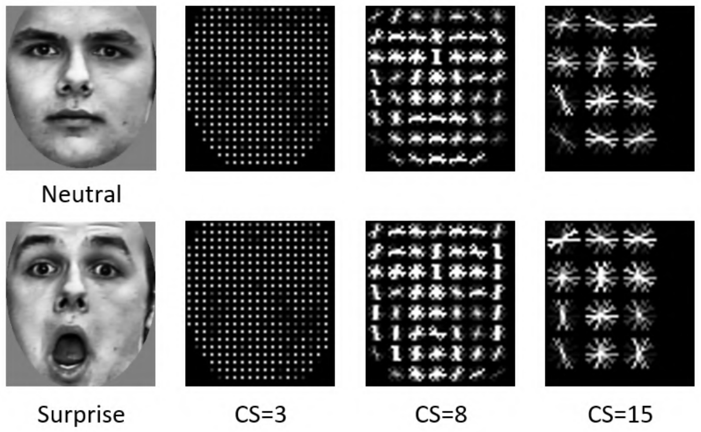


Figure 3.18: Evaluating the HOG parameters visually. CS=Cell Size.

Quantitative Analysis

Using 10-fold cross-validation, several SVM classifiers were trained on HOG feature vectors extracted using different parameter configurations. First, a classifier was trained on the CK+ dataset with 6 expressions. Figure 3.19 reports the average recall performance for varying cell sizes and a varying number of orientations. The results indicate that the highest recall was 98.1% when the cell size was 8, and the number of orientations was 9. A similar performance was obtained when the number of bins was 3. The parameter optimization procedure was repeated using feature vectors extracted from the CK+ subset with 7 expressions, and the results were recorded in Figure 3.20. The cross-validation recall was 96.5% when the number of orientations was 9, and the cell size was 8. This result confirms that these parameters are most suited for FER.

An additional parameter optimization procedure was performed by training SVM classifiers using the RFD dataset to compare with the results obtained from the CK+ datasets. Two subsets of RFD were used. The first dataset contained subjects performing 7 expressions, and the other contained subjects performing 8 expressions. Figures 3.21 and 3.22 give the performance results, showing that the best parameters were a cell size of 8 and the number of orientations of 8. The best recognition rates were reported as 96.4% when classifying 7 expressions and 93.1% when classifying 8 expressions. It must be noted that the result was very similar when the number of orientations was 9, regardless of the number of target expressions. This shows that the performance is still optimal when the number of expressions varies. Tables 3.4 and 3.5 gather the confusion matrix results obtained on the CK+ datasets of 6 and 7 expressions, respectively.

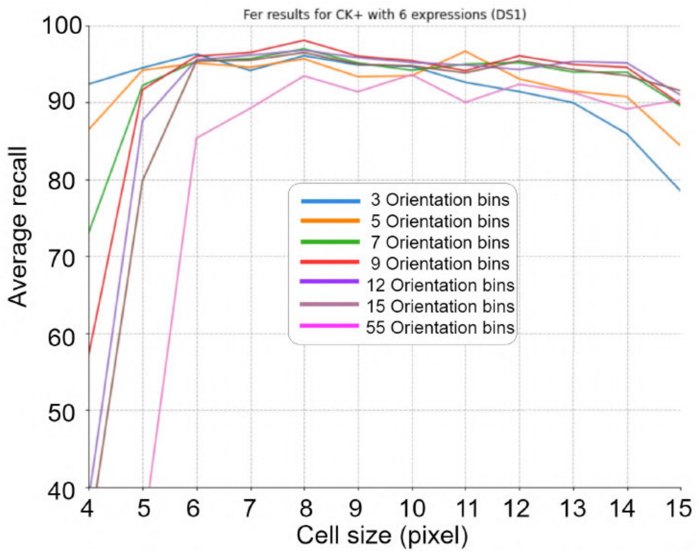


Figure 3.19: FER results for CK+ dataset with 6 expressions.

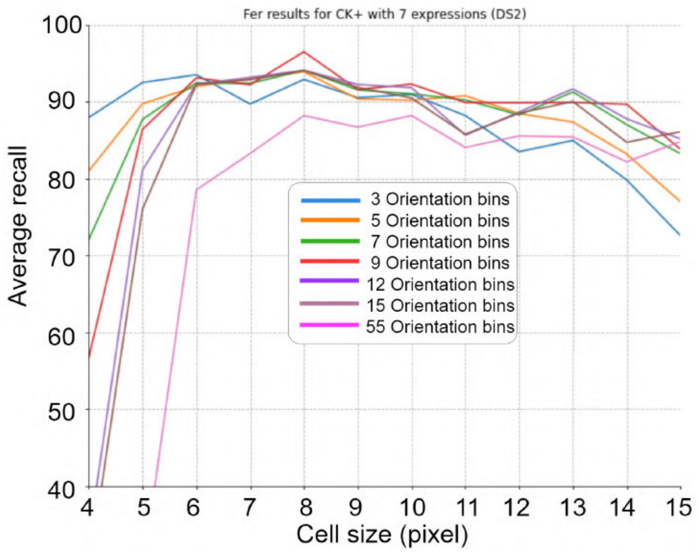


Figure 3.20: FER results for the CK+ dataset with 7 expressions.

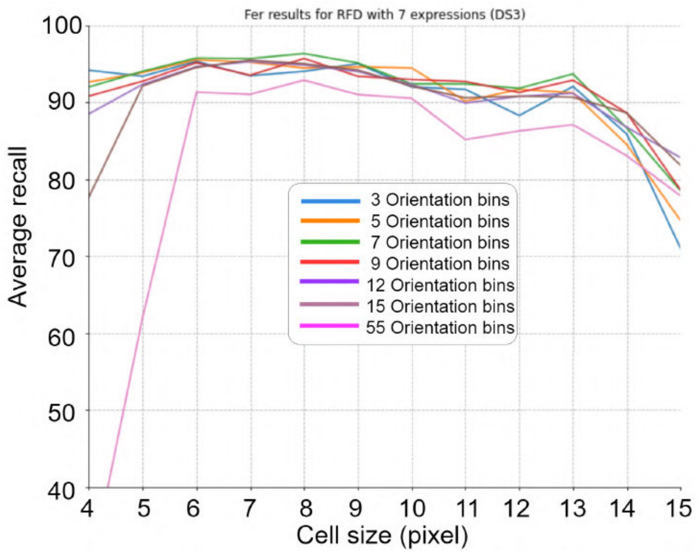


Figure 3.21: FER results for the RFD dataset with 7 expressions.

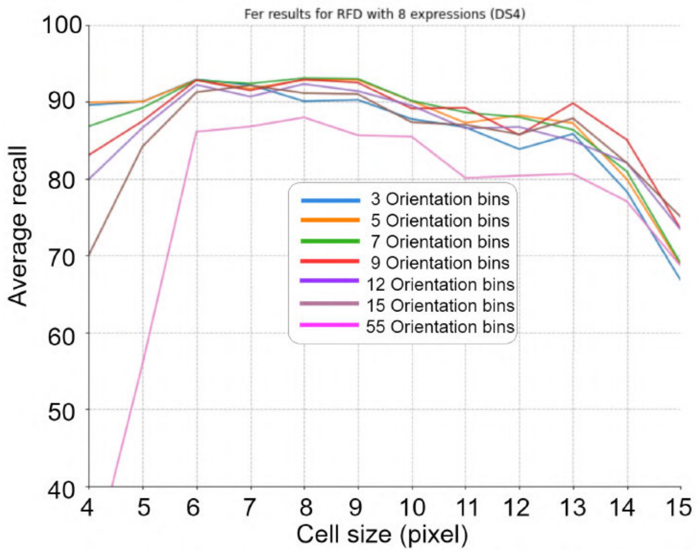


Figure 3.22: FER results for the RFD dataset with 8 expressions.

3.4.2 Experimental Phase 2: Evaluating the Approach on Other Datasets

In this experiment, the classifier was trained using HOG features extracted from the CK+ subset with 6 expressions. Using the best HOG parameters, the following performance was recorded: an average recall of 98.1%, an average precision of 98.4%, an average accuracy of 99.4%, and an average f1-score of 98.0%. Another classifier was trained on faces with 7 expressions achieving the following average performances during cross-validation: average recall of 96.5%, an average precision of 97.2%, an average accuracy of 99.0%, and an average f1-score of 96.6%. Tables 3.6 and 3.7 gather the results obtained when evaluating the approach on the RFD datasets. When the classifier was evaluated on images from the RFD with 7 expressions, the reported results were as follows: an average recall of 96.4%, an average precision of 96.8%, an average accuracy of 99.0%, and an average f1-score of 96.4%. On the other hand, when the classifier was evaluated on images from the RFD with 8 expressions, the reported results are as follows: an average recall of 93.1%, an average precision of 93.7%, an average accuracy of 98.3%, and an average f1-score of 93.0%. It must be pointed out that the classifier's performance depreciates when the number of targets increases. Table 3.8 summarizes the main results obtained on the four datasets.

Table 3.4: FER performance on CK+ with 6 expressions (normalized confusion matrix)

	An	Di	Fe	Ha	Sa	Su
An	93.3	2.2	0	0	4.4	0
Di	1.7	98.3	0	0	0	0
Fe	0	0	100	0	0	0
Ha	0	0	1.4	98.6	0	0
Sa	0	0	0	0	100	0
Su	0	0	0	0	1.5	98.5

An=Anger, Di=Disgust, Fe=Fear, Ha=Happiness, Sa=Sadness, Su=Surprise

Table 3.5: FER performance on CK+ with 7 expressions (normalized confusion matrix)

	Ne	An	Di	Fe	Ha	Sa	Su
Ne	96.7	0	1.7	0	0	1.7	0
An	4.4	88.9	2.2	0	0	4.4	0
Di	1.7	0	96.6	0	1.7	0	0
Fe	2	0	0	98	0	0	0
Ha	0	0	0	1.4	98.6	0	0
Sa	0	0	0	0	0	100	0
Su	2.9	0	0	0	0	1.5	95.6

Ne=Neutral, An=Anger, Di=Disgust, Fe=Fear, Ha=Happiness, Sa=Sadness, Su=Surprise

Table 3.6: FER performance on RFD with 7 expressions (normalized confusion matrix)

	An	Co	Di	Fe	Ha	Sa	Su
An	97	0	0	0	0	3	0
Co	4.5	95.5	0	0	0	0	0
Di	1.5	0	98.5	0	0	0	0
Fe	0	3	0	94	0	0	3
Ha	0	0	0	0	100	0	0
Sa	3	1.5	0	4.5	0	91	0
Su	0	0	0	1.5	0	0	98.5

An=Anger, Co=Contempt, Di=Disgust, Fe=Fear, Ha=Happiness, Sa=Sadness, Su=Surprise

Table 3.7: FER performance on RFD with 8 expressions (normalized confusion matrix)

	An	Co	Di	Fe	Ha	Sa	Ne	Su
An	97	0	0	0	0	0	3	0
Co	3	89.6	0	0	0	7.5	0	0
Di	1.5	0	98.5	0	0	0	0	0
Fe	0	1.5	0	89	0	6	0	3
Ha	0	0	0	6	100	0	0	0
Ne	0	9	0	3	0	82.1	3	0
Sa	1.5	1.5	0	4.5	0	4.5	89.6	0
Su	0	0	0	1.5	0	0	0	98.5

An=Anger, Co=Contempt, Di=Disgust, Fe=Fear, Ha=Happiness, Ne=Neutral, Sa=Sadness, Su=Surprise

Table 3.8: Performance of the proposed approach on different datasets

Dataset	**Precision**	**Recall**	**Accuracy**	**F-score**
CK+6	98.4	98.1	99.4	98.0
CK+7	97.2	96.5	99.0	96.6
RFD7	96.8	96.4	99.0	96.4
RFD8	93.1	93.1	98.3	93.0

3.4.3 Experimental Phase 3: Evaluating Other Descriptors (LBP, CLBP, and WLD)

This section presents the performances of three local descriptors: LBP, CLBP, and WLD. Each local descriptor was evaluated in two stages: in the first stage, feature vectors of a given local descriptor were used to train and evaluate a classifier using 10-fold cross-validation; in the second stage, the computational costs of each descriptor were evaluated. Note that the evaluation of descriptors was performed on the CK+ dataset with 6 expressions.

In the first stage, the parameters for calculating feature vectors were set. For LBP and CLBP descriptors, two parameters must be set: the number of histogram bins and the block size[4]. The values for the number of histogram bins were 8, 16, and 32, while the block sizes were 2 to 24, taking steps of 1. For WLD features, four parameters must be set: T, M, S, and the block size. Block sizes ranged from 2 to 24 (inclusive), while T, M, and S varied independently, taking the values 4, 6, and 8. Figures 3.23, 3.24, and 3.25 show the parameter optimization results for each descriptor. Each graph reports the FER performances (expressed as average recall) using different parameter values. Figure 3.23 shows the FER results for the LBP descriptor using three different bin lengths and seven block sizes. The best recall (95.3%) was produced using 16 bins and a block size of 8×8 pixels. However, the results were the same when using 32 bins and 6×6 pixels blocks. Figure 3.24 gives the parameter optimization results for the CLBP descriptor. It can be observed that the best FER performance (94.6%) was obtained using 8 bins and a block size of 4×4 pixels. Figure 3.25 gives the FER results when using WLD features, keeping M and S fixed, while the block size and T are varied. The best performance (96.7%) was obtained using a block size of 8×8 pixels, $T = 8$, $M = 4$, and $S = 4$. However, note that the performance was very similar when the block size was 7 and 9. Table 3.9 summarizes the best extraction parameters for each descriptor.

The local descriptors were further evaluated using additional performance metrics such as precision, accuracy, and f1-score. Table 3.10 reports the cross-validation metrics obtained for each descriptor. The local descriptors were also evaluated for their computational cost. This evaluation metric has two components: the first measures the time taken in feature extraction, and the other measures the SVM prediction time. Table 3.11 gives the computational costs by taking the average execution time when processing 500 frames. The computational cost of the HOG descriptor was compared to LBP,

[4]In this chapter, block size refers to the side length of a square region in an image.

CLBP, and WLD. The results show that HOG is measurably faster than the other descriptors.

Table 3.9: Best extraction parameters

Descriptor	**No. bins**	**Block size**	**T**	**M**	**S**
LBP	16	8			
CLBP	16	6			
WLD		8	8	4	4

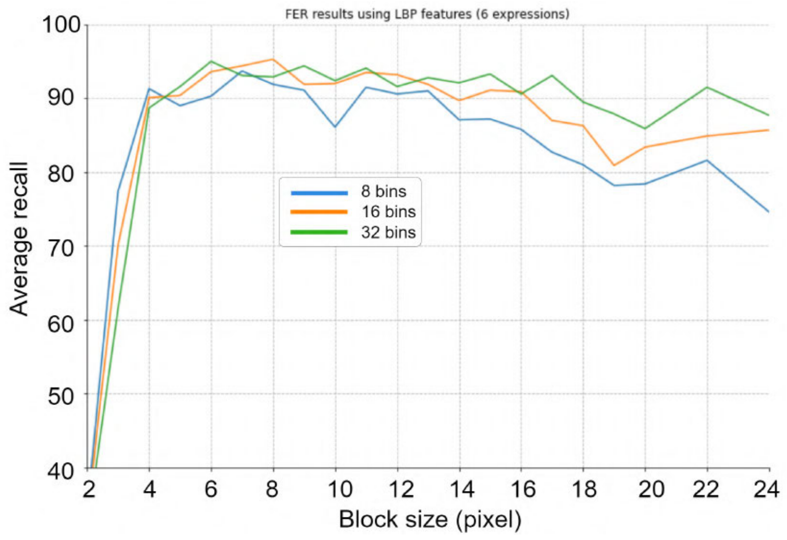


Figure 3.23: FER results using LBP features.

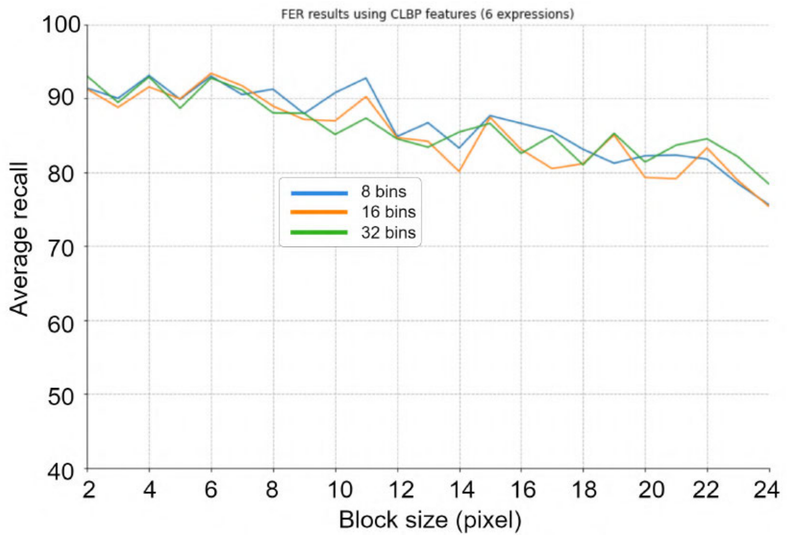


Figure 3.24: FER results when using CLBP features.

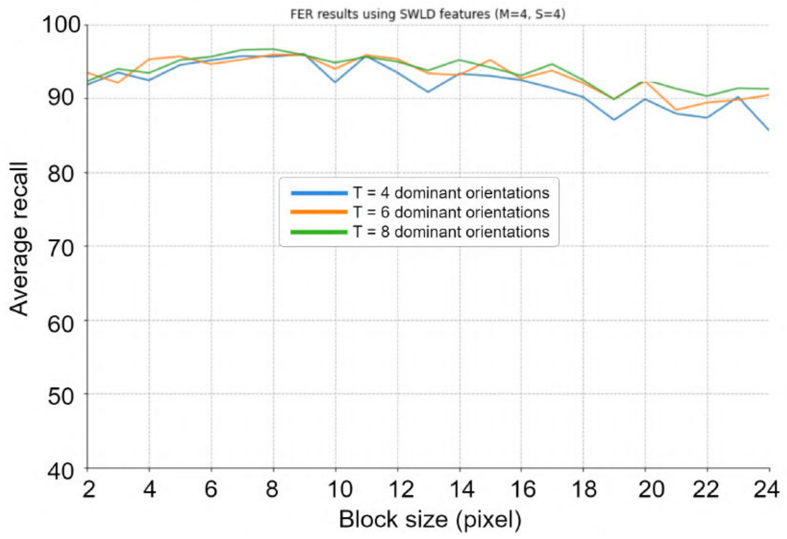


Figure 3.25: FER results using WLD features when $M = 4$, $S = 4$, and the block size is varied. The y-axis reports the average recall while the x-axis reports the block size.

Table 3.10: Cross-validation performances different descriptors

Descriptor	Recall	Precision	Accuracy	F1-score
LBP	95.3	96.4	98.7	95.0
CLBP	93.4	94.6	98.0	93.2
WLD	96.7	97.0	98.9	96.5
HOG	98.1	98.4	99.4	98.0

Table 3.11: Computational cost of HOG compared with other descriptors (in milliseconds)

Descriptor	Detection	Registration	Extraction	Prediction	Total Time
HOG	50.7	179.4	2.4	1.1	233.6
LBP	50.7	179.4	43.3	0.6	274
CLBP	50.7	179.4	115.5	0.8	346.4
WLD	50.7	179.4	170.6	2.5	403.2

3.4.4 Experimental Phase 4: Evaluating SVM on Sequences and Real-Time Data

This section details the results obtained when testing the system on video sequences. The procedure was carried out in two stages: in the first stage, the system was tested on recorded video sequences, and in the second stage, the system was applied to the video output from a laptop's web camera.

Evaluation on CK+ Dataset

After proving the superiority of HOG compared to other texture descriptors, the entire FER system must be evaluated. The system was evaluated quantitatively by running it on sequences from the CK+ dataset. Three sequences were selected for evaluation, and the performance was noted as the images evolved from Neutral to the specified expression. Figure 3.26 gives two examples of the output of the system when two sequences are considered as input. Results reveal that 33 out of 36 images were correctly classified, giving an accuracy of 91.7%. Another way of observing the system's output is by plotting the predicted expressions on a graph. Figure 3.27 gives the graphs plotting the predicted output for six sequences of distinct emotions. The graphs show how the system responds to images changing from the Neutral (Ne) expression to a given expression, e.g., Figure 3.26a shows the expression varying from Neutral to Anger. The graph also shows the initial frame, the first expression change, and the last frame. It is observed that the system correctly predicted most expressions, although there were some classification errors in the sequences related to Anger, Disgust, and Fear.

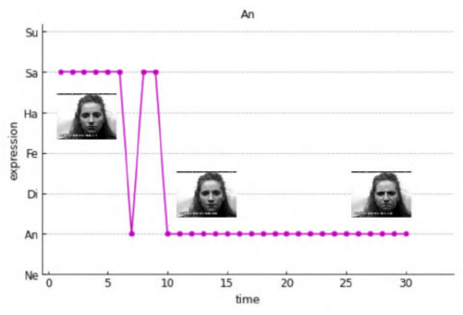
				...					
Real	Ne	Ne	Ne	...	Fe	Fe	Fe	Fe	Fe
Pred	Ne	Ne	Ne	...	Fe	Fe	Fe	Fe	Fe

(a) Sequence S010001

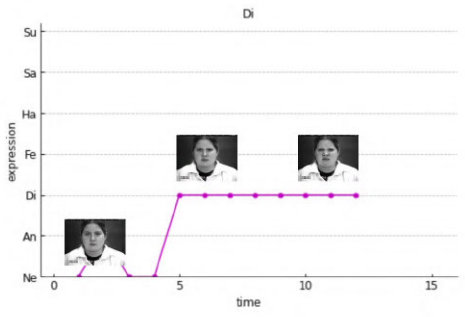
				...					
Real	Ne	Ne	Ne	...	An	An	An	An	An
Pred	Ne	Ne	Ne	...	An	An	An	An	An

(b) Sequence S037003

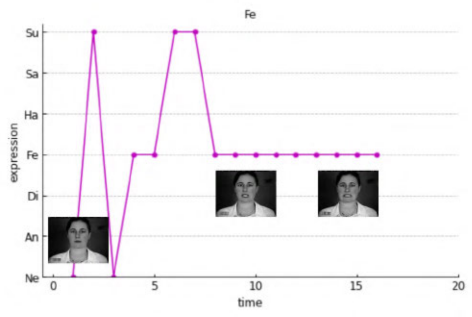
Figure 3.26: Output of the system for the first three images and the last five images for sequences S010001 and S037003. *Ne=Neutral, Fe=Fear, An=Anger*



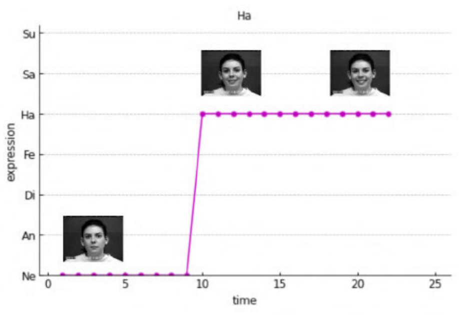
(a) Sequence Anger



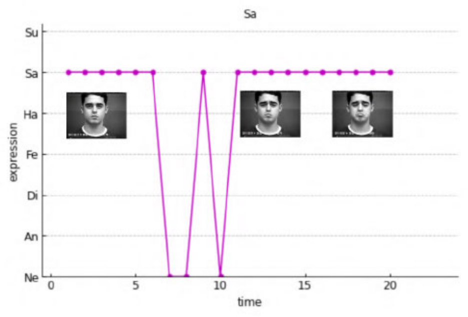
(b) Sequence Disgust



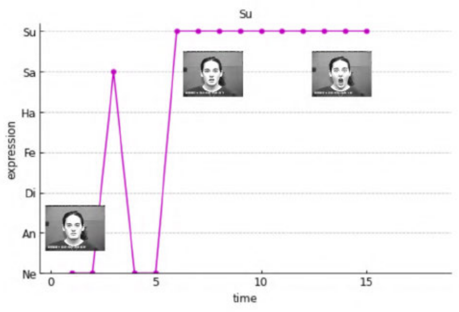
(c) Sequence Fear



(d) Sequence Happiness



(e) Sequence Sadness



(f) Sequence Surprise

Figure 3.27: Graphs of predicted emotions versus time for six sequences.

Evaluation with on-line Video Stream

It was also tested with video input to ensure the system was ready for real-world environments. The expressions were performed by starting with the Neutral expression and ending with the specified expression. In Figure 3.29, it is observed that even though not all frames may be classified the same, the final expression is correct because of the ad-hoc rule. The results also showed that the system could correctly detect and register faces in a cluttered background.

The system was also tested on video input by varying the acquisition angle. Figure 3.28 gives the output when the angle of acquisition is varied between $-30°$ and $30°$, obtaining images in different poses. The system correctly recognized expressions in most poses, showing that the faces were correctly detected and registered. Correctly registered images provide enough information to predict an expression.

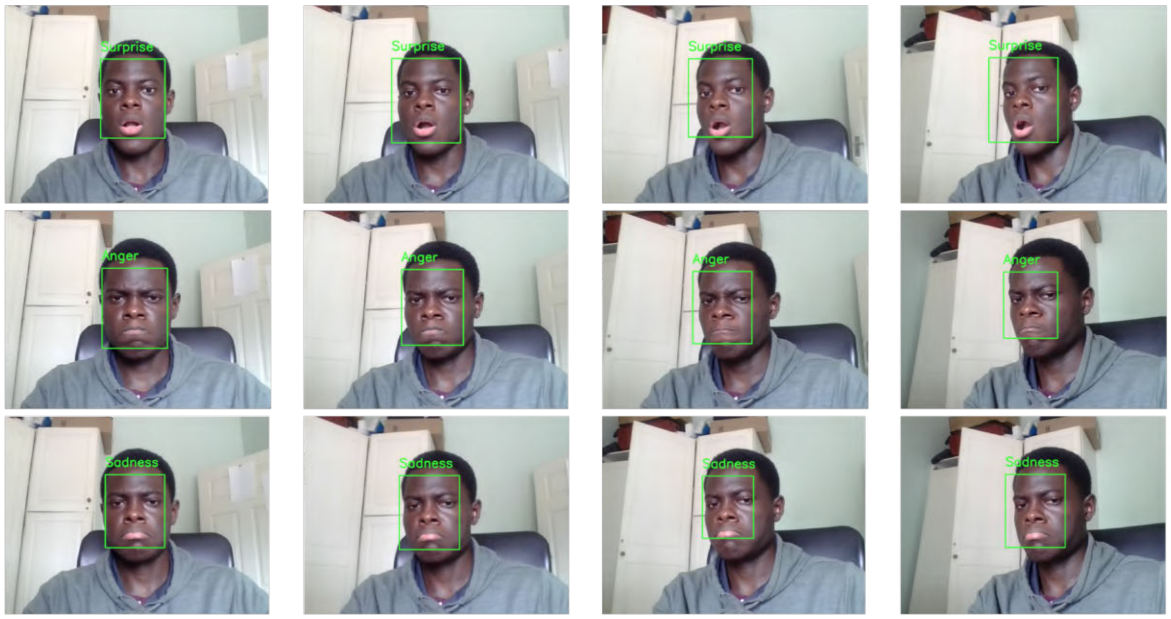


Figure 3.28: Output of the system when the angle of acquisition is varied between $-30°$ and $30°$.

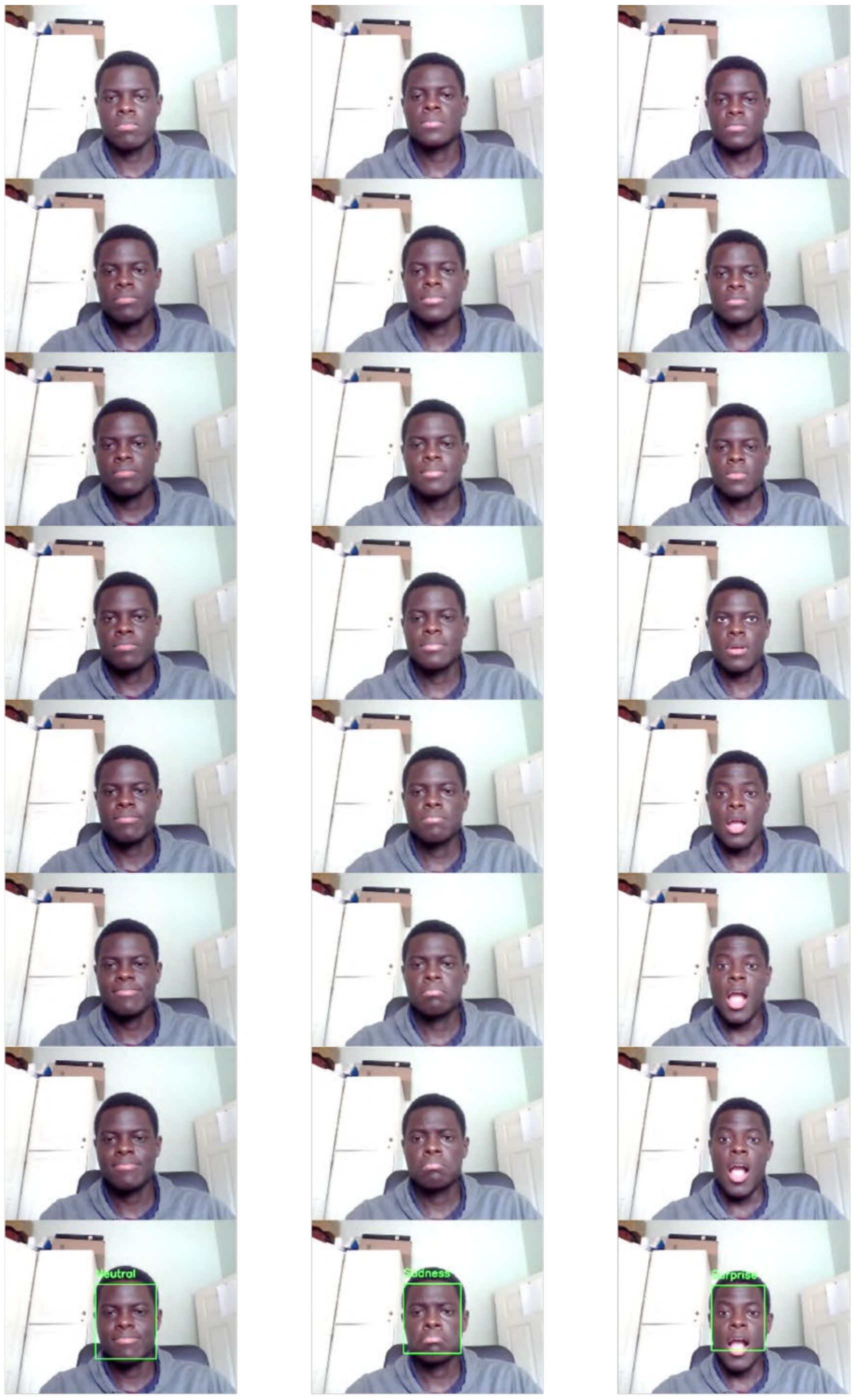


Figure 3.29: Output of the system when tested on a video stream. The images progress from neutral to a specified expression, and the expression is decided when the condition is met (ad-hoc rule).

3.5 Discussion

3.5.1 Factors Affecting FER Performance

The system's performance is seriously affected by three main factors: the selection of datasets, the face registration, and the HOG extraction parameter choice. When the images were selected, a few were found to be incorrectly labeled and had to be replaced. Incorrect labeling tends to introduce ambiguity between expressions, decreasing predictions' accuracy. The registration step is also crucial to increase classification performance. A correctly registered frontal face image must have all the face components to guarantee extraction of the key features and high prediction accuracy.

3.5.2 Comparing HOG with other Descriptors

This section compares HOG with three other descriptors by looking at two factors: the average performance and the computational cost. The recall metric reported during cross-validation is popular when comparing the prediction performance of different FER approaches. Table 3.12 records the average performances of each descriptor when tested on the CK+ dataset with 6 expressions. Results show that the HOG descriptor was superior, with an average recall of 98.1%, followed by the WLD reporting 96.7%. The LBP and CLBP perform less, with average recalls of 95.4% and 93.4%, respectively. More can be said about these descriptors' performance compared with the results obtained in [12]. The authors reported lower performances for all these descriptors and different parameter choices. It is worth noting that the change in extraction parameters caused an improvement in the performances of these descriptors.

The computational costs of each descriptor were also done. Recording the execution times of each descriptor in real time was used to measure their computational cost. The HOG descriptor was significantly more efficient, taking only 2.4ms and 1.0ms during extraction and prediction. This was several magnitudes faster than the other descriptors, which took more than 60ms for extraction or prediction. A probable reason for this is the efficiency of the code used to implement the HOG descriptor compared with the other descriptors. While the HOG descriptor was compiled using Cython[5], the other descriptors were written in ordinary Python code. This means that the source code for the other

[5] `https://pypi.org/project/Cython/` (accessed 1 Jan 2022)

descriptors must also be optimized (i.e., using Cython), and another evaluation performed to get a fairer judgment.

Table 3.12: Average recall performance of HOG compared to other descriptors

HOG	**LBP**	**CLBP**	**WLD**
98.1	95.3	93.4	96.7

3.5.3 Comparison with Existing Approaches

The proposed approach must also be compared to existing state-of-the-art solutions. Table 3.13 gives the average recall values per class when different approaches are evaluated on the CK+ dataset with 6 expressions. The proposed approach was comparable to previous studies and even superior in the classes related to anger, disgust, and surprise.

Table 3.13: Average performances (%) of different approaches when tested on CK+ with 6 expressions

Expression	**[12]**	**Proposed**
Anger	88.6	93.3
Disgust	89	98.3
Fear	100	100
Happiness	100	98.6
Sadness	100	100
Surprise	97.4	98.5
Average	95.8	98.1

3.6 Chapter Summary

This study showed the efficiency of the HOG descriptor and the SVM classifier when developing an automatic FER system. More importantly, an extensive parameter selection was conducted for the HOG descriptor and three other descriptors (LBP, CLBP, and WLD). The superiority of HOG was evidenced in both cross-validation performance and computational cost during real-time execution. The overall system was suitable for real-time applications using temporal analysis. Results show that an appropriate selection of descriptor parameters leads to better performance than what was reported in previous studies.

CHAPTER 4
PARAMETER OPTIMIZATION OF HISTOGRAM-BASED LOCAL DESCRIPTORS FOR FACIAL EXPRESSION RECOGNITION

4.1 Introduction

Feature extraction is one of the critical stages in Facial Expression Recognition (FER) because it must provide the most discriminative features related to a particular expression. The features extracted from a face image must be robust to changes in illumination, pose variation, noise, occlusions, scale variation, and low resolution. The recognition of expressions is also tricky because of the complexity of facial expressions. Local descriptors can solve this problem by capturing the local appearance of image features (such as edges, lines, spots, and textures) related to various facial expressions. Jia et al. [103] provide a solution for image noise, low resolution, and multi-intensity expressions. Hesse et al. [104] handle different face views, while Makhmudkhujaev et al. [47] study the recognition of noisy, and position-varied facial expressions.

This chapter studies four local descriptors that demonstrate robust FER characteristics. This study contributes to developing a method for optimized feature extraction and feature vector calculation. Another significant contribution of this work is showing the importance of correct face registration to preserve only facial features and discard unnecessary regions. The remainder of this chapter is organized as follows.

Section 4.2 presents the background and works related to this research. Section 4.3 discusses the datasets used and the proposed FER approach. Section 4.4 presents the Experimental Results and Discussion, which assesses the impact of face registration and optimizes local descriptors. Section 4.5 concludes the chapter.

4.2 Background and Related Works

Extracting robust features from a face image is one of the most pressing issues for FER systems today. Even the best classifier could fail if the features used do not appropriately represent facial expressions. A robust feature must be highly discriminative, easily computed, of low dimensionality, insensitive to noise, such as illumination changes, and have low intra-class variations [11]. Hence, extensive work has been done to achieve robust feature extraction. Geometric features have been investigated to represent facial expressions [25, 26]. Though geometric features can offer high discriminability, they depend heavily on accurately detecting and aligning fiducial points in a face. Appearance features can solve this problem by extracting features related to the appearance of the entire face or local patches. The approach in [105] uses Gabor features and sparse representation-based classification to recognize facial expressions. The method is robust to illumination changes, different resolutions, and orientations. Sajjad et al. [106] fused two appearance and texture features: HOG and ULTP, and then classified them using the support vector machines. Experiments show that the method performs well in the presence of occlusions.

Local descriptors have proven to be a powerful way of characterizing expression-related information while having low computational complexities. In particular, the histogram-based local descriptors can represent the occurrences of micro-patterns (such as edges, lines, and bumps) that make up facial expressions. However, the aggregation of these micro-patterns into histograms leads to a loss of information about the location of the micropatterns [13, 14]. To address this issue, weighted histogram representations have been proposed to give more weights to regions with higher discriminability (e.g., eyes and mouth) and lower weights to regions with lower discriminability (e.g., image borders). Kabir et al. [7] proposed the LDPv descriptor to characterize facial components' texture and contrast information. The novel descriptor gives weight to the LDP code based on its variance. Recently, a similar approach called the Weighted Statistical Binary Pattern (WSBP) descriptor was developed by Truong et al. [107]. The new descriptor combines the histograms of sign and magnitude components of the mean moments of an image. The histograms are weighted based on a new variance moment that contains distinctive facial features. Though the weighted methods achieve high recognition rates and robustness, they can suffer from high dimensionality and implementation complexity.

A popular technique for enhancing a local description based on a histogram is dividing the face into several overlapping or non-overlapping sub-regions. Shan et al. [13] reviewed FER based on the LBP descriptor by extracting features from sub-regions of a face image. They extracted an LBP histogram with 59 bins from 18×21 pixels sub-regions in a 110×150 image. However, the extraction parameters[1] were not optimized for FER. Carcagni et al. [12] were the first to optimize the cell size and the number of orientations of the HOG descriptor. Results show that the best HOG parameters are independent of the dataset used. The study further optimizes three other local descriptors (i.e., LBP, CLBP, and WLD) and compares them with the HOG descriptor. However, optimizing the other descriptors is not detailed, leaving some questions unanswered, such as whether the optimization of local descriptors is generalizable.

Garcia et al. [38] extract appearance and geometric features around specific face regions. This approach segments three face regions: eye-eyebrows, nose, and mouth, dividing them into sub-regions. The Local Fourier Coefficients (LFC) and Facial Fourier Descriptors (FFD) are extracted from each sub-region, and the feature vectors of all the sub-regions are concatenated. It should be noted that the final feature vector is a linear difference between the expressive and neutral features. Hence, both the expressive and the neutral face are needed to recognize a given expression, which is not always practical in real-life. Also, the approach could be negatively affected by pose variations. Turan and Lam [11] studied the performance of several local descriptors when the resolution and number of sub-regions of the face are varied. Five resolutions were used, and the face was divided into $l \times l$ pixels sub-images where l was varied from 3 to 11. It was reported that higher resolutions and more sub-regions lead to better classification performance. However, the results do not show a trend in recognition performance as the number of sub-regions varies. Also, there is not enough discussion on how the resolution and sub-region size[2] affect the descriptor's ability to represent facial features.

Most of the above approaches extract local descriptors by dividing the face images into sub-regions. However, the approaches do not optimize the extraction parameters for FER, except for a few works which attempt to optimize either the sub-block size, image resolution, histogram bins, or a combination of these [11, 12]. The current work aims to study the importance of optimizing the sub-region size and the number of histogram bins when calculating feature vectors of local descriptors.

[1]The term ”extraction parameters” refers to the variables controlling how a feature vector is calculated.

[2]In this chapter, the terms' sub-region size' and 'block size' are used interchangeably.

4.3 An Approach for Optimized Facial Expression Recognition

A simple and effective method for FER based on local descriptors and conventional classifiers was defined by Slimani et al. [15]. The method involves dividing the face region into several non-overlapping regions and extracting local descriptors from each sub-region. The histogram of each sub-region is calculated separately and concatenated to form a single feature vector for classification. The study reported in this chapter extends Slimani et al.'s [15] approach by optimizing the feature extraction process. Figure 4.1 summarizes the proposed approach for optimized FER, consisting of training and testing phases. In the training phase, images are pre-processed using face detection and registration. Next, a local descriptor is applied to the registered faces to extract facial features. This study examined four local descriptors: Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP), Compound Local Binary Patterns (CLBP), and Weber's Local Descriptor (WLD). Subsequently, feature vectors are generated by dividing the face region into several sub-regions and calculating the histograms of each sub-region. The feature vectors are normalized based on the type of local descriptor used (see Section 4.3.3). To optimize the process of feature vector calculation, the extraction parameters (i.e., the sub-region size and the number of histogram bins) are varied, producing several feature sets. Then, the best extraction parameter settings are found by comparing the classification performances produced by each feature set. Finally, the best feature set is used to train an SVM classifier. In the testing phase, images are processed, and the feature vectors are generated based on the best extraction parameter values obtained from the training phase. After that, the feature vectors are classified as facial expressions using the trained SVM model. Additionally, the SVM classifier is evaluated using 10-fold cross-validation.

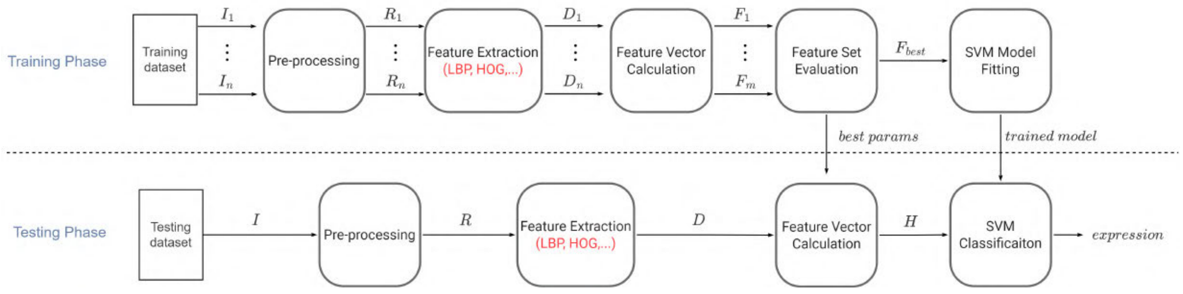


Figure 4.1: The proposed approach for optimized Facial Expression Recognition.

4.3.1 Datasets

The data used in this study was taken from the Extended Cohn Kanade (CK+) dataset and the Radboud Faces Dataset (RFD). The procedure for creating sub-datasets has already been described in Chapter 3. For both datasets, the expressions selected were anger, disgust, fear, happiness, sadness, and surprise. In this study, only datasets with six expressions were used. Table 4.1 summarizes the number of images and expressions considered in this study.

Table 4.1: Summary of datasets

Dataset	**Number of images**	**Number of expressions**
CK+6	347	6
RFD6	402	6

4.3.2 Face Detection and Registration

Faces were detected using a frontal face detector implemented by the OpenCV [91] library for the Python programming language. The faces were then registered, giving them a predefined pose, shape, and size. Detailed discussions of the face detection and registration algorithms have already been presented in Chapter 3 under Section 3.3.1.

4.3.3 Feature Extraction and Feature Vector Calculation

Feature extraction and feature vector calculations are key stages of this FER method. Although the two processes were portrayed separately in the preamble of this section, their workings are very interlinked. The method used in this study divides the face image into several equally sized non-overlapping sub-regions, and a local descriptor is applied to each sub-region, producing several histograms. The final feature vector is generated by normalizing and concatenating the histograms into a single 1D vector. It must be noted that the sub-region histograms are calculated separately and normalized such that the frequencies sum to one[3]. The normalized histograms are then concatenated to

[3]This rule is applied to all the descriptors except the HOG descriptor, where the blocks are normalized in groups using L2-Hys [96].

form the final feature vector. Suppose I_j is a $W_S \times W_S$ matrix of intensities representing the j-th sub-region in the image, and the function $D(x, y)$ computes the descriptor at a position (x, y) in I_j, then the histogram associated with this sub-region is defined by Equation (4.1).

$$H_j = \{h_i\}_{i=0,1,\dots,N_b-1} \tag{4.1}$$

$$h_i = \sum_{x=0}^{W_S-1} \sum_{y=0}^{W_S-1} \delta(D(x, y), i)$$

$$\delta(x, i) = \begin{cases} 1 & \text{if } L(i) \leq x < U(i) \\ 0 & \text{otherwise} \end{cases}, \quad L(i) = i \times \frac{n}{N_b}, \quad U(i) = (i+1) \times \frac{n}{N_b},$$

where n is the number of possible descriptor values and N_b is the number of histogram bins.

The final feature vector is given by Equation (4.2).

$$H = \left\{ \frac{H_0}{(W_S)^2}, \frac{H_1}{(W_S)^2}, \dots, \frac{H_{m-1}}{(W_S)^2} \right\} \tag{4.2}$$

where m is the number of sub-regions.

4.3.4 Feature Set Evaluation

This step involves fitting an SVM model with different feature sets and comparing their classification performances. The performances are ranked by the average accuracies during 10-fold cross-validation. Then, the best feature set and the corresponding best parameter setting are identified. Algorithm 4.1 defines the process of feature set evaluation in more detail.

4.3.5 Facial Expression Classification

In this study, facial expression classification was performed using Support Vector Machines (SVM) classifiers. The SVM hyper-parameters were tuned using the grid-search cross-validation procedure discussed in Chapter 3. Table 4.2 gives the hyper-parameter configuration for each descriptor.

Algorithm 4.1 Feature Set Evaluation

```
Inputs:
    $(F_1, \ldots, F_m)$ : feature sets
    $(p_1, \ldots, p_m)$ : extraction parameters for each feature set
Outputs:
    $F_{best}$ : best feature set
    $p_{best}$ : best extraction parameters
1: $F_{best} \leftarrow F_1$          // Initialize the best feature set
2: $p_{best} \leftarrow p_1$          // Initialize the best parameters
3: $model \leftarrow SVM()$          // Construct a new SVM model
4: $maxScore \leftarrow CrossValidation(model, F_1)$          // Obtain a cross-validation score
5: for $i$ in $(2, 3, \ldots, m)$ do          // Train new models with the remaining feature sets
6:     $model \leftarrow SVM()$
7:     $score \leftarrow CrossValidation(model, F_i)$
8:     if $score > maxScore$ then          // Find the maximum score
9:         $F_{best} \leftarrow F_i$
10:        $p_{best} \leftarrow p_i$
11:        $maxScore \leftarrow score$
12:    end if
13: end for
14: return $F_{best}, p_{best}$
```

Table 4.2: Calibration of the SVM hyper-parameters

Descriptor	**C**	**Gamma**	**Kernel**
HOG	1000	0.05	RBF
LBP	1000	0.05	RBF
CLBP	1000	0.05	Linear
WLD	1000	0.05	Linear

4.4 Experimental Results and Discussion

4.4.1 Experimental Phase 1: The Effect of Face Registration

This experiment investigated the effect of face registration on FER. First, classifiers were trained on feature vectors extracted from unregistered faces, and the performance recognition rates during 10-fold cross-validation were recorded. Then, classifiers were developed on feature vectors from registered faces. The non-registered images were produced by simply detecting a face region and resizing it to 65×65 pixels. In contrast, the registered images were processed using the face registration algorithm and resized to 59×65 pixels to balance performance and feature vector length. The parameter settings used to calculate the feature vectors are given in Table 4.3.

Table 4.3: Extraction parameters setting for comparing registered and unregistered features

Descriptor	**Parameters**
HOG	$\textit{Number of bins} = 9, \textit{block size} = 8$
LBP	$\textit{Number of bins} = 16, \textit{block size} = 7$
CLBP	$\textit{Number of bins} = 8, \textit{block size} = 11$
WLD	$T = 8, M = 4, S = 4, \textit{block size} = 8$

The results in Figure 4.2 show that the registration-based classifiers have a higher recognition rate than the non-registration-based classifiers. The largest increase (22.2% for Anger) is observed for the LBP descriptor. Table 4.4 shows each expression's difference in recognition rate. The results show that registration increases the recognition rate (e.g., fear) of some expressions. In the worst-case scenario, registration does not affect the recognition performance (e.g., HOG happiness). The largest increase can be observed for the LBP descriptor for the anger expression. Observing the results for the various descriptors, the following pattern is observed: the performances are improved by the registration process regardless of the descriptor used. These results suggest that face registration has a positive (and non-negligible) impact on the feature extraction phase and, ultimately, recognition performance. Figure 4.3 shows an example of a registered image compared to a non-registered image.

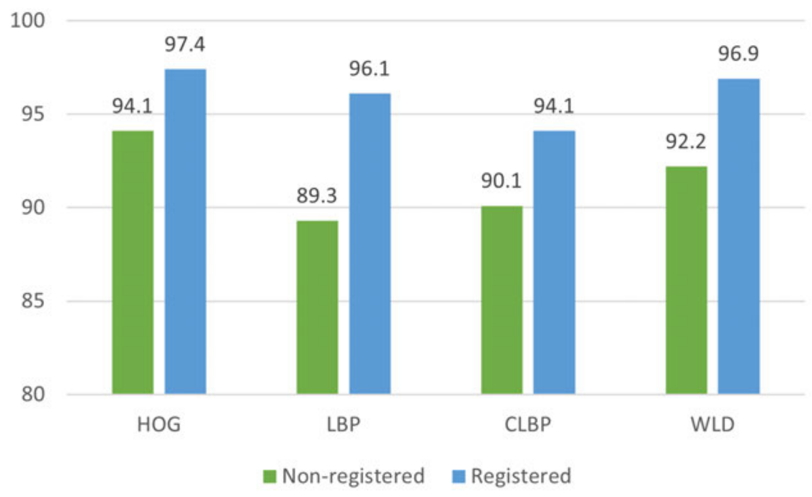


Figure 4.2: Recognition rates obtained with and without registered faces.

Table 4.4: Recognition performance per class before and after registration

Expression	HOG		LBP		CLBP		WLD	
	NR[1]	RE[2]	NR	RE	NR	RE	NR	RE
Anger	88.9	93.3	68.9	91.1	84.4	80	77.8	93.3
Disgust	91.5	96.6	94.9	93.2	83.1	94.1	93.2	94.9
Fear	92.0	100	90.0	100	96.0	98.0	92.0	98.0
Happiness	98.6	98.6	95.7	98.6	91.3	97.1	95.7	98.6
Sadness	96.4	98.2	91.1	96.4	91.1	96.4	98.2	98.2
Surprise	98.5	98.5	97.1	97.1	95.6	98.5	95.6	98.5
Average	94.1	97.4	89.3	96.1	90.1	94.1	92.2	96.9

[1] NR=Non-registered, [2] RE=Registered

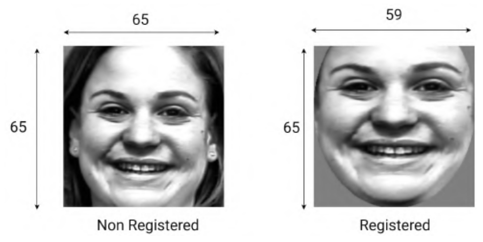


Figure 4.3: Example of non-registered and registered images.

4.4.2 Experimental Phase 2: Optimizing Local Descriptors

This experiment aimed to find the most optimal way of extracting four local descriptors: HOG, LBP, CLBP, and WLD. First, different feature sets were created based on the selected local descriptor. Then, 10-fold cross-validation was done using each feature set, and the average recalls was recorded. The feature sets were created by varying the extraction parameters of each descriptor as follows. The parameters for LBP and CLBP were the number of histogram bins and the block size. The number of bins varied between 8, 16, and 32, and the block size varied from 2 to 24, taking steps of 1. For the HOG descriptor, the number of orientations varied between 3, 5, 7, 9, 12, 15, and 55, and the cell size varied between 4 and 15 (inclusive). Concerning WLD, four parameters were tuned: T, M, S, and the block size. Block sizes ranged from 2 and 24 (inclusive) while T, M and S took values of 4, 6, and 8.

Figures 4.4, 4.5, 4.6, and 4.7 gather the FER results (expressed as average recall) when each descriptor was optimized on the CK+ dataset of 6 expressions. The best parameters for extracting HOG descriptor are reported as 9 orientations and a cell size of 8, giving a recognition rate of 97.4%. Concerning the LBP features, the recognition rate has a maximum value of 96.1% when the number of bins is 16 and the block size is 7 pixels. The best recognition rate for the CLBP descriptor is 94.1% when the parameters are 8 bins and 11-pixel blocks. Finally, the best performance (96.9%) for WLD is obtained using the following parameters: block size = 8, $T = 8$, $M = 4$, and $S = 4$. The same parameter optimization procedure was repeated; the features are extracted from the RFD dataset of 6 expressions. The results in Figure 4.8 show that the best parameters for extracting HOG were 7 orientations and a cell size of 8 (RR[4]=97.2%). From Figure 4.9, the best parameters of LBP parameters are reported as 16 bins and a block size of 8 (RR=94.6%). The best parameters for CLBP are 8 bins and 6-pixel blocks, as shown in Figure 4.10 (RR=95.6%). Finally, Figure 4.11 shows the best parameters of WLD as the block size of 7, $T = 6$, $M = 4$, and $S = 4$ (RR=96.5%). The main observation is that the recognition rate rises and then falls as the block size varies. This suggests that smaller blocks lead to too much fine-grain information, which is less helpful in representing action units. On the other hand, larger blocks may merge multiple action units, which is less valuable because it mixes discriminatory and non-discriminatory information. A similar finding was presented in [14].

The above results proved that the performance of these local descriptors improved due to optimizing extraction parameters. However, the parameters must be assessed to determine whether they depend on the training dataset used. Tables 4.5 and 4.6 give the extraction parameters and feature vector length for each descriptor on each dataset. The results show that the most frequent value for block size is 8; however this varies from 6 to 8. This means that a block size of 8×8 pixels could be the best window for extracting features at the current resolution. On the other hand, the number of histogram bins is different for all descriptors. These findings reveal that the optimum set of parameters depends on the training dataset, which is not ideal.

[4]RR=Recognition Rate

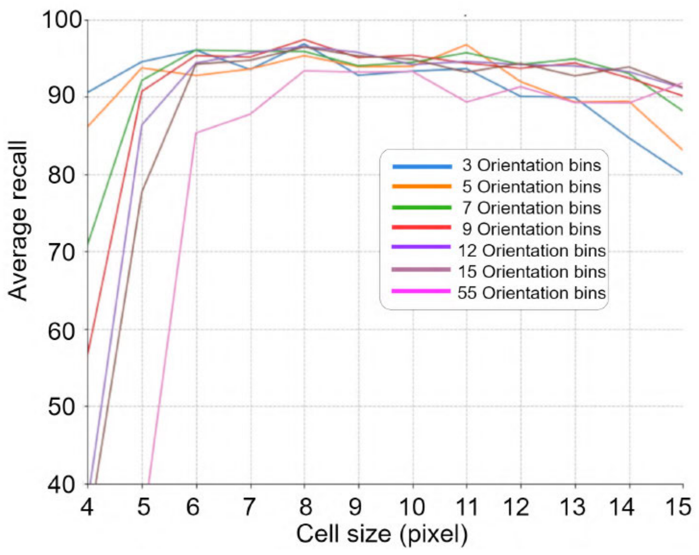


Figure 4.4: HOG optimization results (CK+ with 6 expressions).

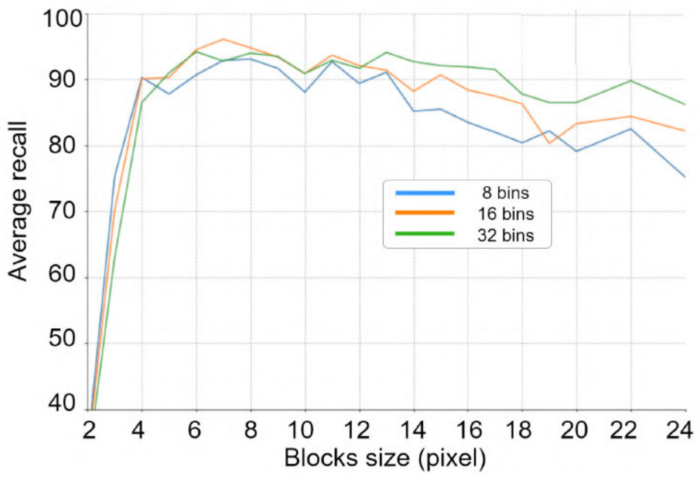


Figure 4.5: LBP optimization results (CK+ with 6 expressions).

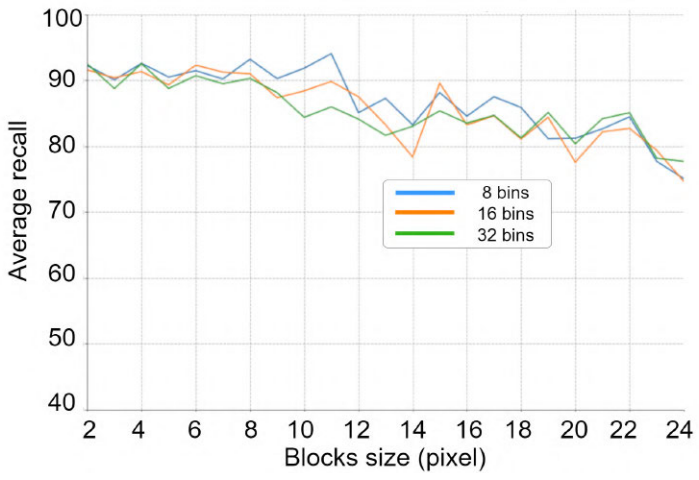


Figure 4.6: CLBP optimization results (CK+ with 6 expressions).

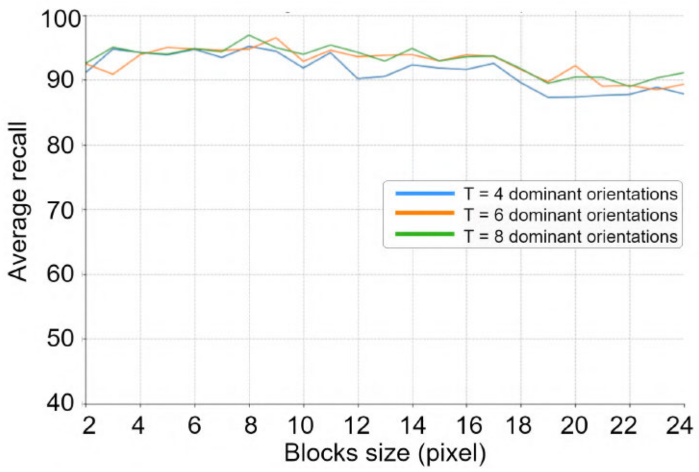


Figure 4.7: WLD optimization results (CK+ with 6 expressions).

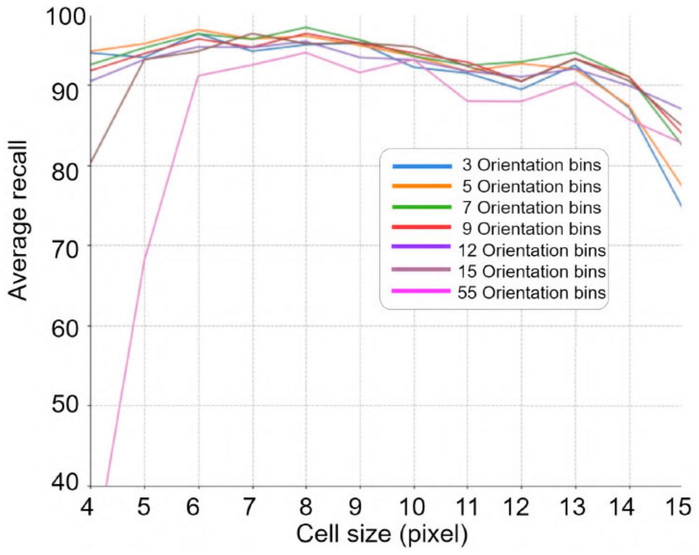


Figure 4.8: HOG optimization results (RFD with 6 expressions).

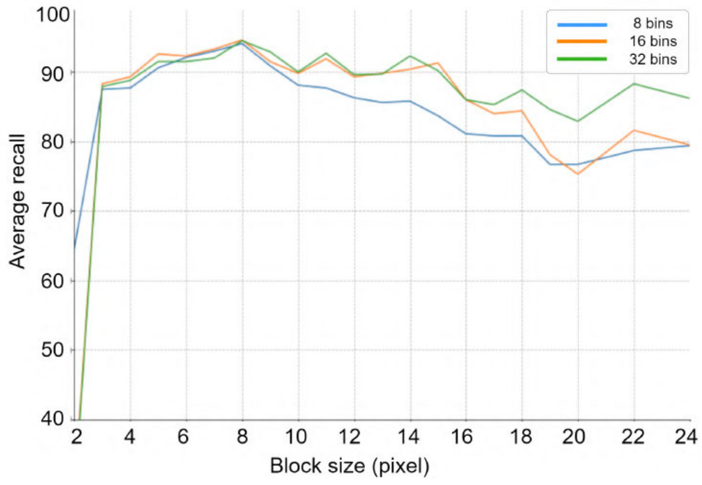


Figure 4.9: LBP optimization results (RFD with 6 expressions).

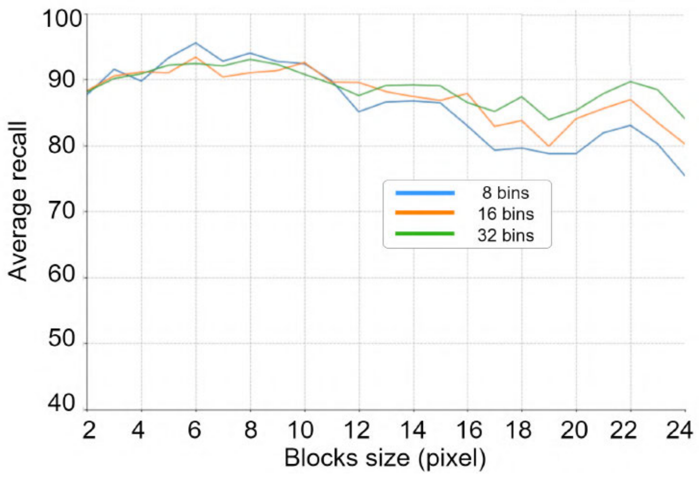


Figure 4.10: CLBP optimization results (RFD with 6 expressions).

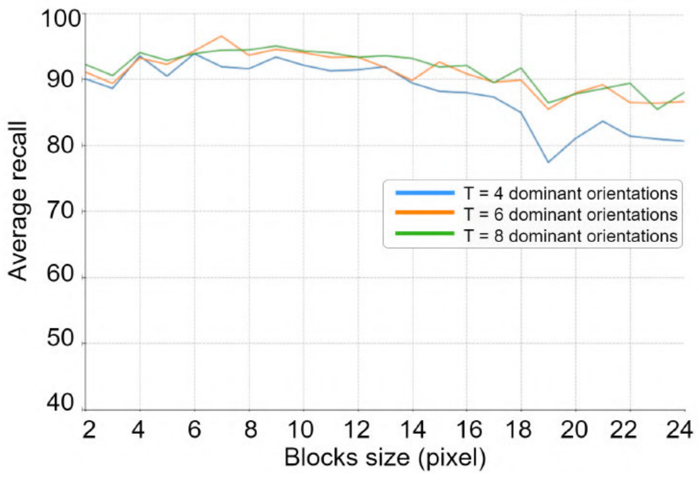


Figure 4.11: WLD optimization results (RFD with 6 expressions).

Table 4.5: Best parameters for the extraction on the CK+ dataset

Descriptor	Number of bins	Block size	T	M	S	Feature vector length
HOG	9	8				2430
LBP	16	7				1440
CLBP	8	11				576
WLD		8	8	4	4	9216

Table 4.6: Best parameters for the extraction on the RFD dataset

Descriptor	Number of bins	Block size	T	M	S	Feature vector length
HOG	7	8				1890
LBP	16	8				1152
CLBP	8	6				1760
WLD		7	6	4	4	8640

4.4.3 Comparison with the State-of-the-Art

Comparing FER systems in the literature is challenging because many studies do not report on their evaluation strategy. Also, the training-testing data split often varies from one study to another. Nevertheless, the result from this study can be fairly compared to works using 10-fold cross-validation and the recognition rate[5] as a performance metric. Table 4.7 gives various recent state-of-the-art approaches evaluated on the extended Cohn Kanade (CK+) dataset[6]. For each approach, the recognition rates reported by the authors were recorded. All the approaches used SVM for classification, showing superior performance, especially when working with small datasets. The best recognition rate was achieved by the proposed approaches using the HOG descriptor improving, outperforming the results in [12] by a difference of 1.6%. Compared to the same source, the proposed approach gives a better

[5]The recognition rates were obtained from the authors of the methods.

[6]When a value is not available, it is replaced by '-'

recognition rate for LBP, CLBP and WLD descriptors. This improvement is probably due to a more optimum selection of descriptor parameters in this study.

Table 4.7: Performance of the proposed approach compared to various state-of-the-art approaches

Approach	**Year**	**Feature**	**Feature vector length**	**Classifier**	**Registration**	**Number of images**	**Recognition rate (%)**
[7]	2012	LDPv	2352	SVM	No	1224	96.7
[12]	2015	HOG	-	SVM	Yes	347	95.8
[12]	2015	LBP	-	SVM	Yes	347	91.7
[12]	2015	CLBP	-	SVM	Yes	347	92.3
[12]	2015	WLD	-	SVM	Yes	347	86.5
[19]	2018	CLGDNP	-	SVM	No	1482	95.3
[18]	2019	DRADAP	-	SVM	No	1043	90.6
[8]	2019	LBP	2478	SVM	Yes	150	96.2
Proposed	2022	HOG	2430	SVM	Yes	347	97.4
Proposed	2022	LBP	1440	SVM	Yes	347	96.1
Proposed	2022	CLBP	576	SVM	Yes	347	94.1
Proposed	2022	WLD	9216	SVM	Yes	347	96.9

4.4.4 Factors Affecting the Recognition Performance

The system's performance is seriously affected by three main factors: the selection of datasets, the face registration, and the feature extraction parameters. Incorrect labeling also tends to introduce ambiguity between expressions and can decrease predictions' accuracy. The registration step is crucial to increase classification performance. A correctly registered frontal face image must have all the facial components to guarantee extraction of the key features and high prediction accuracy. This finding was also pointed out in [12], and the results confirm that the recognition performance is higher when using registered images than when the images are not registered.

4.5 Chapter Summary

This chapter presented a study on the optimization of local descriptors for FER. The proposed approach recognized facial expressions in three stages: face detection and registration, feature extraction, and classification. Findings reveal that face registration is critical for improving recognition performance. The study was validated on two popular facial expression datasets, and the performance obtained was comparable to the state-of-the-art. Parameter optimization was performed for four descriptors: HOG, LBP, CLBP, and WLD, resulting in improved recognition rates. Although the best parameters' values differed depending on the dataset used, the block sizes were very similar, taking values close to 8×8 pixels. The results showed that the performance peaked at this point. Considering this, if improved, the current optimization procedure could be used to develop future FER systems based on local descriptors.

CHAPTER 5
A COMPARATIVE STUDY OF LOCAL DESCRIPTORS AND CLASSIFIERS FOR FACIAL EXPRESSION RECOGNITION

5.1 Introduction

A few attempts have been made to compare the performances of local descriptors and conventional classifiers applied to Facial Expression Recognition (FER). In 2018, Turan and Lam [11] studied 27 local descriptors under different conditions, such as varying image resolutions and the number of sub-regions. Two classifiers were used to recognize expressions on four facial expression databases. Their results were comparable to the state-of-the-art deep learning approaches on well-known datasets. Slimani et al. [15] studied the independent performance of 46 LBP variants for FER. A single classification technique (i.e., SVM) was used to classify seven expressions. The study showed that several local descriptors, initially proposed for other classification problems, outperformed the state-of-the-art four databases. In both studies mentioned above, only one or two classifiers were considered, whereas more classifiers still need to be investigated.

The work presented in this chapter compares the performances of six local descriptors and four machine learning for FER. Multiple experiments were conducted under different settings, such as varied extraction parameters, different numbers of expressions, and two datasets, to discover the best combinations of local descriptors and classifiers for FER. The rest of this chapter is organized as follows: Section 5.2 reviews local descriptors and classifiers, Section 5.3 gives the Materials and Methods, Section 5.4 presents the Experimental Results and Discussion, and Section 5.5 concludes the report.

5.2 Related Works

5.2.1 Local Binary Patterns

Since its introduction as a textured-based descriptor in 1996, the Local Binary Patterns (LBP) descriptor has undergone several improvements to produce a more robust descriptor. For example, Shan et al. [13] improved LBP by dividing the face image into sub-regions of various sizes and positions. The AdaBoost algorithm was used to learn the most discriminative LBP histograms. The so-called Boosted-LBP features were then classified as expressions using machine learning techniques such as template matching, Support Vector Machines, Linear Discriminant Analysis, and linear programming. LBPTOP (Local Binary Patterns on three Orthogonal planes) extends LBP to encode temporal changes and spatial changes [108]. This method extracts LBP patterns from three orthogonal planes: the spatial, vertical spatio-temporal plane, and horizontal spatio-temporal plane. In [90], a new kernel-based manifold learning method called kernel discriminant isometric mapping is proposed to reduce the dimension of LBP features. The features were then classified using the Nearest Neighbor classifier. Recently, Guo et al. [109] proposed the Extended Local Binary Patterns on three Orthogonal planes (ELBPTOP) for spontaneous micro-expressions recognition, and in [110], a smaller LBP feature vector that is also resistant to noise is introduced. The novel descriptor considers four neighbors and diagonal neighbors separately, and an adaptive window and averaging in radial directions to improve the feature extraction. Finally, the feature vector is classified using SVM.

5.2.2 Local Directional Patterns

Local Directional Patterns (LDP) are binary patterns obtained by encoding the edge responses in a local neighborhood of an image [53]. The LDP descriptor computes eight directional edge responses at each pixel and encodes the responses as an 8-bit binary code using the relative strengths of the edge responses. Specifically, the edge responses are computed by applying Kirsch edge masks. To form a binary code, the k most significant responses are set to 1 while the remaining $8 - k$ bits are set to 0. In summary, the LDP code is defined by Equation (5.1).

$$LDP_k = \sum_{j=0}^{7} B(m_j - m_k)2^j \tag{5.1}$$

where m_j is the j-th directional response, m_k is the k-th most significant directional response and $B(x)$ is defined as

$$B(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

5.2.3 Angled Local Directional Patterns

Recently, the Angled Local Directional Patterns (ALDP) descriptor was proposed to improve upon the LDP descriptor by addressing two drawbacks of LDP: (1) the static choice of the most significant bits and (2) the value of the center pixel was ignored [111]. To address these issues, the ALDP descriptor is extracted by first generating Kirsch mask responses $(m_0, \ldots, m_7)$ just like LDP, then computes angular vector components, $(p_0, \ldots, p_7)$ in four angles (0, 45, 90, 135). The angular vector components are defined by Equations (5.2)−(5.9).

$$p_0 = b(m_0 - c, m_4 - c) \tag{5.2}$$

$$p_1 = b(m_3 - m_2, m_1 - m_2) \tag{5.3}$$

$$p_2 = b(m_7 - m_6, m_5 - m_6) \tag{5.4}$$

$$p_3 = b(m_7 - m_0, m_1 - m_0) \tag{5.5}$$

$$p_4 = b(m_6 - c, m_2 - c) \tag{5.6}$$

$$p_5 = b(m_3 - m_4, m_5 - m_4) \tag{5.7}$$

$$p_6 = b(m_5 - c, m_1 - c) \tag{5.8}$$

$$p_7 = b(m_3 - c, m_7 - c) \tag{5.9}$$

where c is the center pixel of a 3×3 grid and $b(r, s)$ is defined as

$$b(r,s) = \begin{cases} 1 & \text{if } r \geq 0 \text{ and } s \leq 0 \\ 0 & \text{otherwise} \end{cases}$$

Consequently, the ALDP code of pixel is defined by Equation (5.10).

$$ALDP_{x,y}(p_0, p_1, \ldots, p_7) = \sum_{i=0}^{7} p_i \times 2^i \tag{5.10}$$

where (x,y) are the coordinates of a pixel in an input image. Finally, a histogram of ALDP codes is used as a feature vector for classification. One of the drawbacks of ALDP is the large size of the feature vector. Since it considers eight angular vector components, the basic histogram size is 256, unlike LDP, which has 56 histogram bins.

5.2.4 K-Nearest Neighbors

K-Nearest Neighbors (KNN) is one of the simplest machine learning algorithms successfully applied to the FER problem. Sohail and Bhattacharya [112] proposed an approach based on eleven feature points representing the principal muscle actions and the KNN classifier to recognize six basic facial expressions, achieving an average accuracy of 90.76%. Panchal and Pushpalatha [113] proposed a new descriptor combining LBP and Asymmetric Region LBP for feature extraction and KNN for classification. When classifying seven expressions, an overall recognition accuracy of 95.1% was achieved on the JAFFE database. It should be noted that the KNN algorithm does not construct a general internal model but simply stores training data instances. Hence, no explicit training is required. Hence, there are zero training costs. However, finding the k neighbors becomes computationally expensive when classifying high-dimensionality features.

5.2.5 Multi-Layer Perceptron

Multi-Layer Perceptron (MLP) neural network has shown encouraging performances in recognizing emotions. In [31], a system is proposed for classifying facial expressions using the Viola-Jones algorithm for face detection, the HOG descriptor for feature extraction, PCA for dimensionality reduction, and MLP for classification. They reported an average accuracy of 82.97% on the CK+ database when classifying eight basic expressions. Boughrara et al. [114] presented a new constructive training algorithm for the Multi-Layer Perceptron. Unlike most traditional algorithms, which fix the structure of the neural network before training, the proposed constructive training algorithm learns the network architecture and performs the learning process simultaneously. The approach uses the Perceived Facial Images (PFI) in eight directions to extract features from a face image. The predicted expression is obtained by a fusion of the neural networks corresponding to the eight directions. The main drawback of MLP is that it is often time-consuming and tedious to determine a proper structure to guarantee convergence and avoid over-fitting.

5.3 Materials and Methods

5.3.1 Method

Chapter 4 presented an optimized Facial Expression Recognition approach using local descriptors and the SVM classifier. This study adapts the optimized FER approach to investigate several other local descriptors and multiple classifiers instead of one. Figure 5.1 gives the modified FER approach for this study, which is segmented into two phases: a training phase and a testing phase. In the training phase, images are processed using face detection and face registration. Then, feature vectors are extracted from the registered face by applying a local descriptor and calculating histograms. The local descriptors considered for this study are Local Binary Patterns (LBP), Compound Local Binary Patterns (CLBP), Local Directional Patterns (LDP), Angled Local Directional Patterns (ALDP), Weber's Local Descriptor (WLD), and Histogram of Oriented Gradients (HOG). As in Chapter 4, various feature sets are generated by varying the extraction parameters, and the best feature set is used to train a classifier. However, this study considers multiple classifiers; thus, the extraction parameter settings may differ depending on the classifier used.

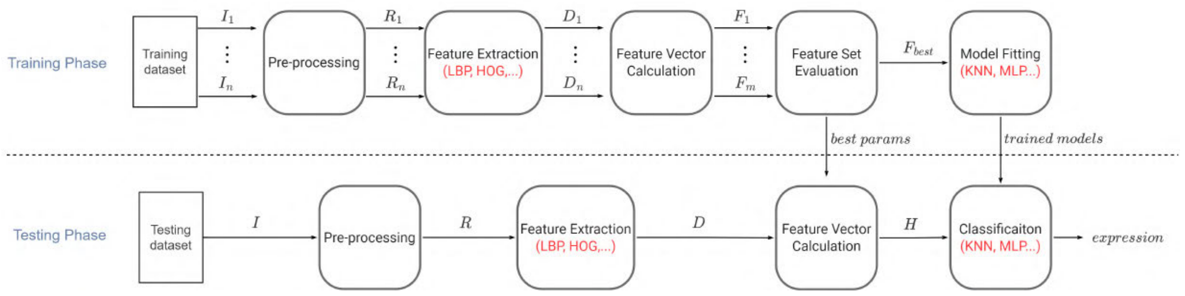


Figure 5.1: FER method combining local descriptors and classifiers.

In the testing phase, test images are processed, and feature vectors are classified as expressions. Again, the main difference between the current approach and the one in Chapter 4 is that more than one classifier is used here. Another distinction in this approach is that it allows the number of expressions to vary between 6, 7, and 8. The following classifiers were used: Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Naive Bayes (NB), and Multi-layer Perceptron (MLP). Table 5.1 gives the hyper-parameter settings related to each classifier.

5.3.2 Datasets

These experiments used the Extended Cohn Kanade (CK+) and Radboud Faces Dataset (RFD). These datasets have been presented in detail in previous chapters (see Chapter 3). Table 5.2 summarizes the number of samples and expressions in each dataset.

Table 5.1: Classifier hyper-parameters

Classifiers	**Parameters**
SVM	Kernel = RBF, C = 1000, Gamma = 0.05
Gaussian NB	Automatically selected
KNN	k = 50
MLP	solver = Adam, learning rate = 0.001, number of passes over the training data = 200

Table 5.2: summary of the datasets

Database name	Number of images	Number of classes
CK+6	347	6
CK+7	407	7
RFD6	402	6
RFD7	469	7
RFD8	536	8

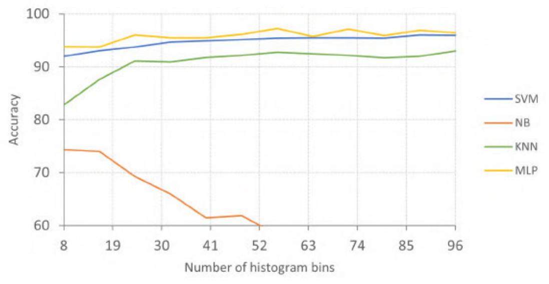
5.4 Experimental Results and Discussion

5.4.1 Performance Analysis for Varying the Number of Histogram Bins

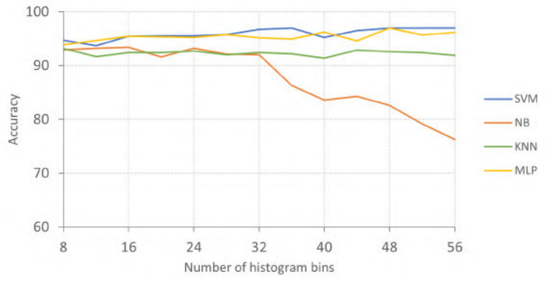
In this experiment, the face regions were first detected and aligned using a face registration algorithm that automatically positions the eyes and crops on the face image to discard background pixels. The aligned face images were then resized to a resolution of 100×100 pixels. By setting the size of the sub-regions to 13×13 pixels (i.e., 49 sub-regions in total), feature vectors are extracted from each image. The number of histogram bins was varied to reduce the feature vector length while achieving a high recognition rate. Note that for WLD, the histogram bins were set as $M = 2$, $S = 2$, and T was varied from 2 to 24. Figure 5.2 gives the results[1] obtained when the number of histogram bins is varied for different descriptor and classification combinations. Different trends can be observed depending on the type of classifier used. When SVM and MLP classifiers are used, the classification performance improves or stays the same as the bin number increases. A similar result is seen in the KNN classifier. However, the large size of the features negatively affects recognition rates of the considered local descriptors (e.g., with the LDP descriptor, the performance decreases as the number of bins increases). On the other hand, when using the NB classifier, the recognition rates decrease as the number of bins rises, regardless of the type of features used. Tables 5.3 and 5.4 give the results,

[1]Results are given as the average accuracy during 10-fold cross-validation.

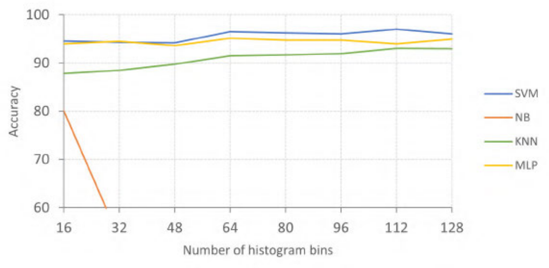
with the optimum accuracies highlighted in bold.



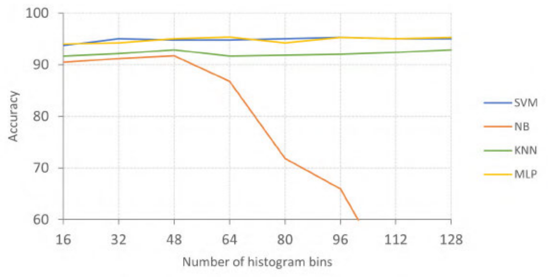
(a) WLD results



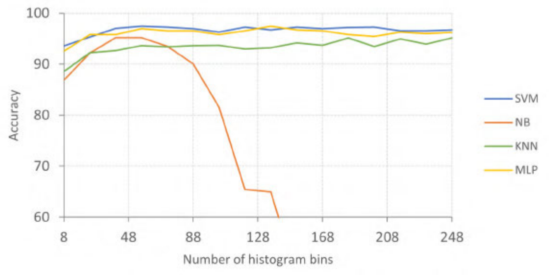
(b) LDP results



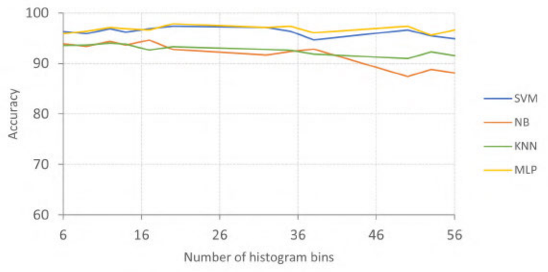
(c) ALDP results



(d) LBP results



(e) CLBP results



(f) HOG results

Figure 5.2: FER results for different number of bins on RFD dataset with 6 expressions.

Table 5.3: FER results on RFD with 6 expressions for different number of bins (WLD, LDP and ALDP)

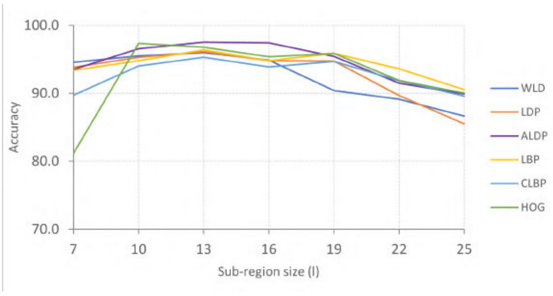
	WLD						LDP						ALDP				
Bins	8	16	80	88	96		8	12	48	52	56		16	32	96	112	128
SVM	92.0	93.0	95.4	**96.0**	96.0		94.7	93.7	**96.9**	96.9	96.9		94.6	94.3	96.0	**97.0**	96.0
NB	**74.4**	74.0	55.9	50.8	57.7		92.9	**93.2**	82.6	79.2	76.2		**80.2**	53.7	35.2	36.2	32.1
KNN	82.8	87.6	91.7	92.0	**93.0**		**93.1**	91.7	92.6	92.4	91.9		87.9	88.5	91.9	**93.0**	92.9
MLP	93.8	93.7	95.9	**96.9**	96.4		93.8	94.6	**96.9**	95.7	96.1		94.0	94.5	94.8	94.0	**95.0**

Table 5.4: FER results on RFD with 6 expressions for different number of bins (LBP, CLBP and HOG)

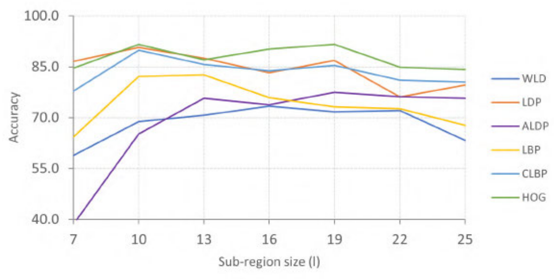
	LBP						CLBP						HOG				
Bins	16	32	96	112	128		8	24	216	232	248		6	9	50	53	56
SVM	93.7	95.0	**95.3**	95.0	95.0		93.6	95.3	96.5	96.5	**96.7**		96.3	95.9	**96.6**	95.5	94.9
NB	90.5	**91.2**	65.9	47.3	38.8		86.9	**92.2**	43.0	42.0	45.3		**93.9**	93.4	87.4	88.8	88.1
KNN	91.7	92.2	92.1	92.4	**92.9**		88.6	92.3	95.0	94.0	**95.2**		93.6	**93.7**	91.0	92.3	91.6
MLP	94.0	94.2	95.2	95.0	**95.3**		92.5	95.8	**96.3**	96.0	96.2		95.9	96.4	**97.4**	95.6	96.6

5.4.2 Performance Analysis for Varying the Sub-region Size

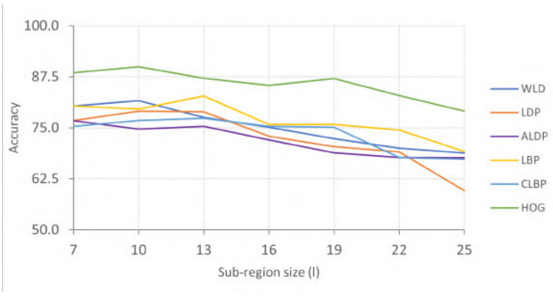
Figure 5.3 shows the results for varying the sub-regions size on the CK+ dataset, while Figure 5.4 gives the results from the RFD dataset. Similarly, Tables 5.5, 5.6, 5.7, and 5.8 highlight the best results in bold for each classifier and descriptor combination. A sub-region can be represented by a square of size $l \times l$, where l varies from 7 to 25, taking steps of 3. As observed in the tables, the larger the sub-regions size (and the smaller the number of sub-regions), the better the recognition rates. This trend is observed for all the classifier and descriptor combinations. It is also worth noting that no single sub-region size works best for all descriptor and classifier combinations. However, the results suggest that using sub-regions of 10$\times$10 and 13$\times$13 pixels leads to some of the highest recognition rates. Table 5.9 gives the extraction parameters selected for the subsequent series experiments based on the results obtained from the two previous experiments (see Sections 5.4.2 and 5.4.1).



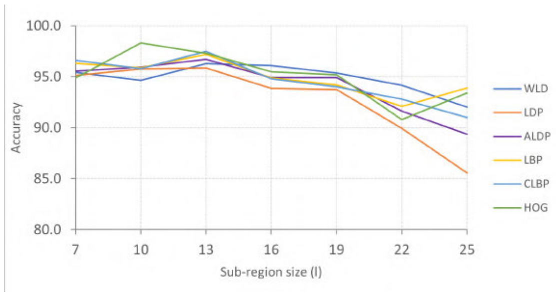
(a) SVM results



(b) NB results

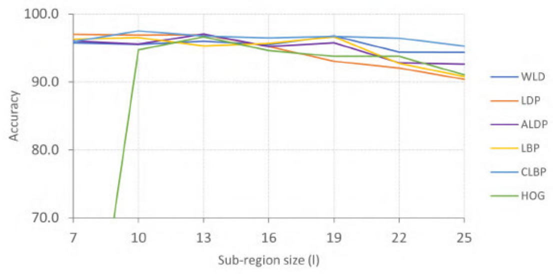


(c) KNN results

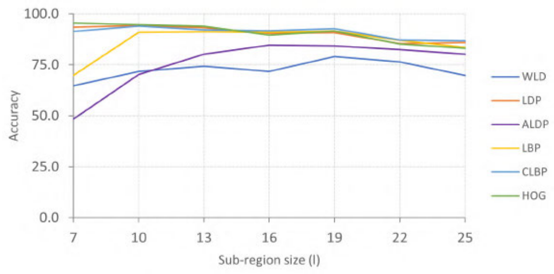


(d) MLP results

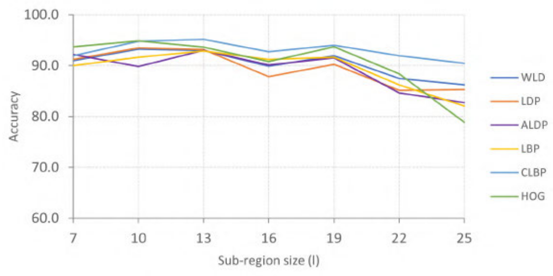
Figure 5.3: FER results for different sub-regions sizes on CK+ with 6 expressions.



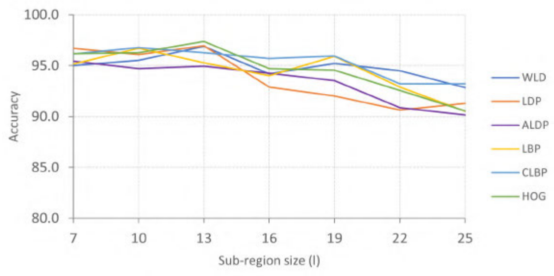
(a) SVM results



(b) NB results



(c) KNN results



(d) MLP results

Figure 5.4: FER results for different sub-regions sizes on RFD with 6 expressions.

Table 5.5: FER results for different sub-regions sizes on CK+ with 6 expressions using 10-fold (SVM and NB)

	SVM								NB						
$l =$	7	10	13	16	19	22	25		7	10	13	16	19	22	25
WLD	94.6	95.5	**96.0**	94.9	90.4	89.1	86.6		58.9	68.9	70.7	**73.4**	71.7	72.1	63.3
LDP	93.8	95.3	**96.1**	94.8	94.8	89.7	85.5		86.6	**90.7**	87.5	83.3	86.9	76.1	79.6
ALDP	93.5	96.6	**97.5**	97.4	95.4	91.5	89.9		38.6	65.2	75.7	73.8	**77.5**	76.2	75.7
LBP	93.4	94.8	**96.4**	94.8	95.9	93.6	90.6		64.3	82.1	**82.7**	75.9	73.2	72.6	67.7
CLBP	89.7	94.0	**95.3**	93.9	94.7	91.9	89.6		77.8	**89.9**	85.6	83.8	85.4	81.1	80.6
HOG	81.2	**97.4**	96.8	95.4	95.9	91.9	90.0		84.6	**91.6**	87.1	90.2	91.6	84.9	84.1

Table 5.6: FER results for different sub-regions sizes on CK+ with 6 expressions using 10-fold (KNN and MLP)

	KNN								MLP						
$l =$	7	10	13	16	19	22	25		7	10	13	16	19	22	25
WLD	80.3	**81.7**	77.6	75.1	72.4	70.0	68.8		95.4	94.7	**96.3**	96.1	95.4	94.2	92.0
LDP	76.8	**79.0**	78.9	72.9	70.4	69.1	59.6		95.1	**95.8**	95.9	93.9	93.7	89.9	85.6
ALDP	**76.7**	74.7	75.4	72.0	68.9	67.7	67.6		95.6	95.9	**96.7**	94.9	94.9	91.6	89.4
LBP	80.4	79.6	**82.7**	75.8	75.8	74.4	69.2		**96.3**	95.9	97.2	94.9	94.2	92.1	93.9
CLBP	75.4	76.8	**77.3**	75.3	75.1	67.7	67.3		**96.6**	95.8	97.5	94.8	94.0	92.8	91.0
HOG	88.5	**90.0**	87.1	85.4	87.0	82.9	79.1		94.9	**98.3**	97.3	95.5	95.2	90.8	93.4

Table 5.7: FER results for different sub-regions sizes on RFD with 6 expressions using 10-fold (SVM and NB)

	SVM								NB						
$l =$	7	10	13	16	19	22	25		7	10	13	16	19	22	25
WLD	95.8	95.5	**96.0**	95.5	96.0	94.4	94.4		64.7	71.7	74.4	71.7	74.0	**76.5**	69.8
LDP	96.9	**97.0**	96.9	95.2	93.0	92	90.4		93.5	**94.4**	93.2	90.5	90.8	85.3	85.9
ALDP	96.0	95.6	**97.0**	95.2	95.8	92.8	92.6		48.4	70.2	80.2	**84.6**	84.2	82.4	80.2
LBP	96.3	**96.5**	95.3	95.7	96.0	92.7	90.8		69.8	90.9	**91.5**	90.9	91.4	87.0	83.6
CLBP	95.9	**97.5**	96.7	96.5	96.7	96.4	95.3		91.4	**94.0**	92.2	91.7	92.7	87.1	86.7
HOG	28.9	94.7	**96.6**	94.6	93.8	93.8	91.0		**95.5**	94.7	93.9	89.6	91.5	85.2	83.1

Table 5.8: FER results for different sub-regions sizes on RFD with 6 expressions using 10-fold (KNN and MLP)

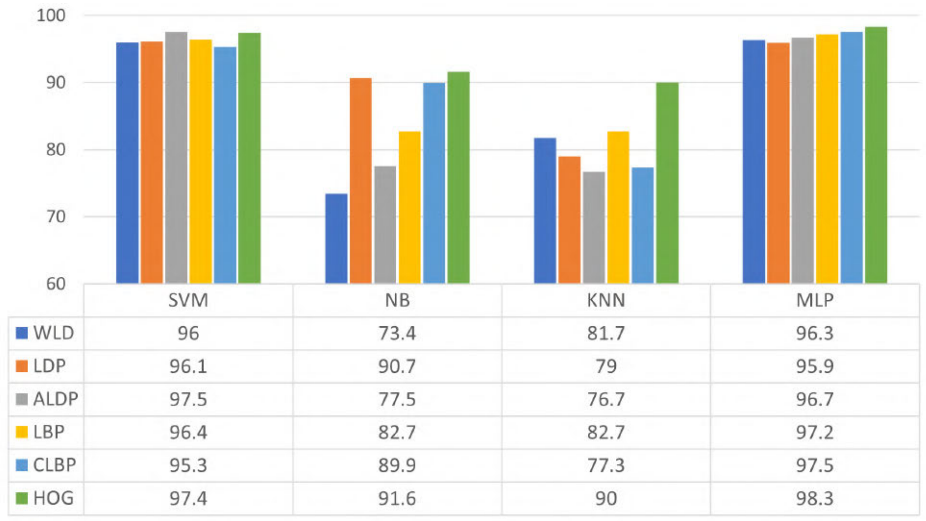
	KNN								MLP						
$l =$	7	10	13	16	19	22	25		7	10	13	16	19	22	25
WLD	91.0	**93.3**	93.0	89.9	91.9	87.5	86.2		95.0	95.5	**96.9**	94.2	95.2	94.5	92.9
LDP	91.2	**93.5**	93.1	87.8	90.2	85.2	85.3		96.7	96.1	**96.9**	92.9	92.0	90.6	91.3
ALDP	92.2	89.8	**93.0**	90.1	91.5	84.6	82.7		94.7	94.7	**95.0**	94.2	93.5	90.9	90.2
LBP	90.0	91.7	**92.9**	91.2	91.6	86.2	82.1		95.2	**96.7**	95.3	94.0	95.9	92.9	90.5
CLBP	91.9	94.8	**95.2**	92.7	94.0	91.9	90.5		96.2	**96.8**	96.3	95.7	96.0	93.2	93.2
HOG	93.7	**94.9**	93.7	90.8	93.7	88.4	78.8		96.2	96.3	**97.4**	94.7	94.6	92.6	90.6

Table 5.9: Best extraction parameters for each combination of classifier and descriptor expressed as (number of bins, block size)

	SVM	NB	KNN	MLP
WLD	(88, 19)	(8, 19)	(96, 10)	(88, 13)
LDP	(48, 7)	(12, 10)	(8, 10)	(48, 13)
ALDP	(112, 13)	(16, 16)	(112, 13)	(128, 7)
LBP	(96, 19)	(32, 19)	(128, 13)	(128, 10)
CLBP	(248, 10)	(24, 10)	(248, 13)	(216, 10)
HOG	(50, 13)	(6, 7)	(9, 10)	(50, 13)

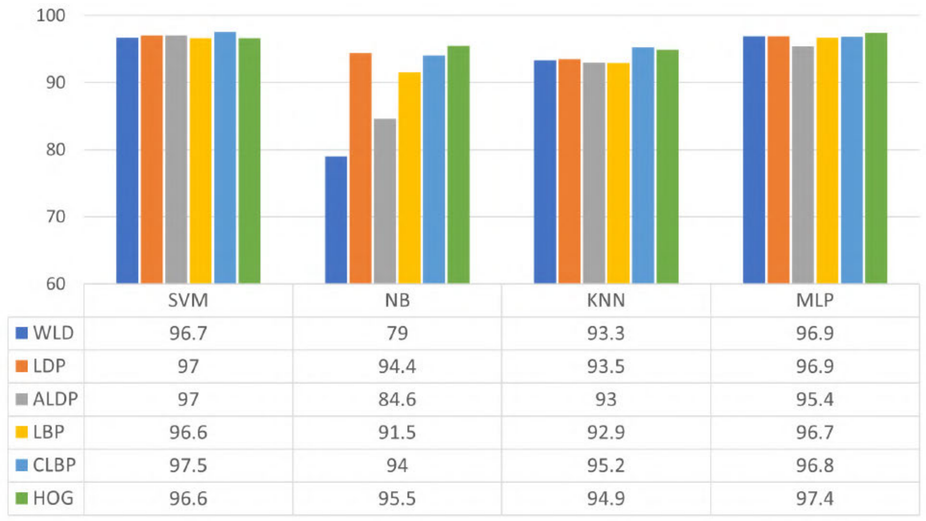
5.4.3 Performance Analysis of the Classifiers

Figure 5.5 summarizes the best results when varying the sub-region sizes on the CK+ dataset (6 expressions). The results show that the best recognition rates ($95 - 98\%$) were achieved using SVM and MLP classifiers, while the lowest accuracies were produced by the NB and KNN classifiers ($73 - 91\%$). A similar trend can be observed on the RFD dataset (6 expressions), as seen in Figure 5.6. The performances of the SVM and MLP classifiers were further analyzed by repeating the previous experiments across two datasets with a varying number of expressions. For the CK+ dataset, the expressions were Anger, Disgust, Surprise, Sadness, Happiness, and Disgust in the 6-class problems, and the 7-class problem added the Neutral expression to the previous list. For the RFD dataset, the expressions were Anger, Disgust, Surprise, Sadness, Happiness, and Disgust in the 6-class problem, the 7-class problem added the expression of Contempt to the previous list, and the 8-class problem added the Neutral expression. Considering the 6-class problem, SVM and MLP achieve similar classification rates (above 95%) on the two datasets. Although their performances are similar, the MLP classifier achieved slightly better recognition rates than SVM when classifying WLD, ALDP, LBP, CLBP, and HOG descriptors. Both classifiers are negatively affected by the introduction of the Neutral. The most significant performance drop is seen in the ALDP + SVM combination, where the accuracy drops by almost 7%. The recognition rates of both classifiers do not change much when the Contempt expression is introduced in the RFD dataset with seven expressions. Table 5.10 gives the results for the best combinations when the number of expressions varies from 6 to 8. It can be observed that HOG + SVM and HOG + MLP have the highest recognition rates.



	SVM	NB	KNN	MLP
🟦 WLD	96	73.4	81.7	96.3
🟧 LDP	96.1	90.7	79	95.9
⬜ ALDP	97.5	77.5	76.7	96.7
🟨 LBP	96.4	82.7	82.7	97.2
🟦 CLBP	95.3	89.9	77.3	97.5
🟩 HOG	97.4	91.6	90	98.3

Figure 5.5: FER results for the best of sub-regions on the CK+ dataset (6 expressions).



	SVM	NB	KNN	MLP
🟦 WLD	96.7	79	93.3	96.9
🟧 LDP	97	94.4	93.5	96.9
⬜ ALDP	97	84.6	93	95.4
🟨 LBP	96.6	91.5	92.9	96.7
🟦 CLBP	97.5	94	95.2	96.8
🟩 HOG	96.6	95.5	94.9	97.4

Figure 5.6: FER results for the best of sub-regions on the RFD dataset (6 expressions).

Table 5.10: Comparison of recognition rates on the CK+ and RFD datasets with varying number of expressions

	CK+			**RFD**		
Descriptor + Classifier	**6-class**	**7-class**		**6-class**	**7-class**	**8-class**
WLD + SVM	96.0	92.4		96.7	95.9	91.5
LDP + SVM	96.1	93.3		97.0	96.8	94.0
ALDP + SVM	97.5	91.2		97.0	96.8	94.1
LBP + SVM	96.4	93.0		96.6	97.0	93.0
CLBP + SVM	95.3	93.0		97.5	97.0	93.5
HOG + SVM	97.4	93.4		96.6	97.0	93.8
WLD + MLP	96.3	94.2		96.9	95.9	91.9
LDP + MLP	95.9	93.2		96.9	96.2	93.4
ALDP + MLP	96.7	92.7		95.4	95.3	93.0
LBP + MLP	97.2	93.1		96.7	96.4	94.0
CLBP + MLP	97.5	93.5		96.8	97.4	93.1
HOG + MLP	98.3	95.1		97.4	96.4	94.0

5.4.4 Performance Analysis per Class

In this Section, only the best-performing classifiers are analyzed with respect to their per-class performances. Table 5.11 gives the results obtained on the CK+ datasets. The best combination is HOG + MLP, exhibiting an accuracy of 100% for Happiness, Sadness, and Surprise, while the lowest accuracy is 95.6% when for Anger. The following best combination is ALDP + SVM, showing the lowest accuracy when classifying Anger and the highest accuracy when classifying Fear and Happiness. Results reveal that the Anger expression is the most challenging expression to classify, with recognition rates as low as 82.2% in combinations like CLBP + SVM and LDP + MLP. The results show that both SVM and MLP struggle to recognize the Anger expression from the others. This could be due to ambiguity between classes or wrong labeling. Table 5.12 shows that the best combinations are CLBP + SVM and HOG + MLP, with similar results on the RFD dataset. In both cases, the highest accuracies are obtained when classifying Anger, Happiness, and Surprise. CLBP + SVM performs slightly better than HOG + MLP when classifying the Sadness expression, but HOG + MLP is better at classifying Fear. In summary, HOG + MLP achieved the best results on both datasets.

Table 5.11: FER recognition rates per class on the CK+ dataset

Descriptor + Classifier	**Anger**	**Disgust**	**Fear**	**Happiness**	**Sadness**	**Surprise**	**Accuracy (%)**
WLD + SVM	86.7	91.5	100	100	98.2	100	96.1
LDP + SVM	86.7	93.2	100	100	96.4	100	96.1
ALDP + SVM	93.3	94.9	100	100	98.2	98.5	97.5
LBP + SVM	86.7	96.6	100	97.1	98.2	100	96.4
CLBP + SVM	82.2	94.9	100	97.1	98.2	100	95.4
HOG + SVM	91.1	94.9	98.0	100	100	100	97.3
WLD + MLP	84.4	98.3	98.0	97.1	100	100	96.3
LDP + MLP	82.2	96.6	100	98.6	98.2	100	95.9
ALDP + MLP	82.2	96.6	98.0	98.6	98.2	100	95.6
LBP + MLP	86.7	94.9	96.0	98.6	100	98.5	95.8
CLBP + MLP	86.7	94.9	98.0	98.6	98.2	98.5	95.8
HOG + MLP	95.6	96.6	98.0	100	100	100	98.4

Table 5.12: FER recognition rates per class on the RFD dataset

Descriptor + Classifier	**Anger**	**Disgust**	**Fear**	**Happiness**	**Sadness**	**Surprise**	**Accuracy (%)**
WLD + SVM	98.5	100	92.5	100	91.0	98.5	96.8
LDP + SVM	97.0	98.5	94.0	100	95.5	97.0	97.0
ALDP + SVM	98.5	98.5	91.0	100	95.5	98.5	97.0
LBP + SVM	95.5	100.0	95.5	100	94.0	95.5	96.8
CLBP + SVM	100	98.5	92.5	100	95.5	98.5	97.5
HOG + SVM	100	98.5	94.0	100	91.0	97.0	96.8
WLD + MLP	100	98.5	94.0	100	92.5	97.0	97.0
LDP + MLP	98.5	98.5	91.0	100	95.5	98.5	97.0
ALDP + MLP	97.0	100	88.1	100	91.0	97.0	95.5
LBP + MLP	98.5	100	94.0	100	92.5	95.5	96.8
CLBP + MLP	98.5	98.5	94.0	100	92.5	97.0	96.8
HOG + MLP	100	98.5	94.0	100	94.0	98.5	97.5

5.4.5 Analyzing the Computational Costs

The computational costs of all the considered descriptors and classifiers were obtained by measuring the processing times during extraction and prediction. First, the extraction time[2] of each descriptor was measured by setting the sub-region size to 13×13 pixels and the number of bins to 56. Then, the number of histogram bins of each descriptor was set to the optimal value based on the classifier used (see Table 5.9), and the sub-region size was set to 13×13 pixels. Tables 5.13 and 5.14 give the extraction and prediction times, respectively, averaged over 500 iterations. Results show that the best extraction time was achieved by WLD (0.54s) followed by HOG (1.72s). The longest times are seen in the LBP and CLBP descriptors. NB was the fastest among the considered classifiers, with an average prediction time of 197ms, followed by MLP, which took an average of 438ms. SVM came in the third

[2]Extraction time refers to both feature extraction and feature vector calculation.

position, and KNN was the last.

The training times of classifiers when trained with different descriptors were also measured. For all considered descriptors, the sub-region size was set to 19×19 pixels, and the number of histogram bins[3] was set to 48. The features were extracted from the RFD dataset with six expressions. Table 5.15 shows that KNN and NB had the fastest training time (a few milliseconds), SVM, which took 29ms. MLP was the slowest, taking up to 6 seconds during training. All the experiments were performed on Windows 10 PC with an AMD Ryzen 7 (3700U) processor and 12GB of RAM.

Table 5.13: Extraction times (in seconds)

WLD	**LDP**	**ALDP**	**LBP**	**CLBP**	**HOG**
0.54	3.79	3.9	5.69	12.49	1.72

Table 5.14: Prediction times of various classifiers (times are given in milliseconds)

	WLD	LDP	ALDP	LBP	CLBP	HOG	Average
SVM	1300	530	1240	1050	2520	1970	1435
NB	150	240	180	200	200	210	197
KNN	3010	500	3260	3560	6360	1160	2975
MLP	310	140	460	460	680	580	438

[3]Or the number of orientations in case of the HOG descriptor

Table 5.15: Training times of various classifiers (times are given in milliseconds)

	WLD	LDP	ALDP	LBP	LBP	HOG	Average
SVM	41	30	22	26	39	14	29
NB	9	6	6	8	6	4	6
KNN	1	1	1	1	1	1	1
MLP	6520	5680	7489	5991	6234	2836	5792

5.4.6 Comparison with the State-of-the-Art

Several local descriptors have been proposed in recent years to tackle the FER problem [115, 116, 31]. Table 5.16 gives the performances of various existing FER systems compared to the proposed best combinations of local descriptors and classifiers. The table shows that several local descriptors perform as well as deep neural networks. Furthermore, Shokrani et al. [117] proposed the Pyramid Histogram of Oriented Gradients (PHOG) descriptor and KNN classifier, achieving an average accuracy of 100% on the CK+ dataset. Results like these are difficult to compare with other approaches because they depend on the selection of the train and testing images. Recent studies evaluate their methods using 10-fold cross-validation because it offers a more reliable measure of classification performance. In k-fold cross-validation, each sample is used for training and testing once. On the other hand, not all samples are available during the evaluation of train-test splits. Hence, a high accuracy during a 67%-33% split, as in [117], is not always a good performance indicator.

It is clear from the table that the proposed approaches using HOG and CLBP either perform better or as efficiently as other approaches. Although some existing approaches have produced higher accuracies, they generally produce larger feature vectors. The proposed method finds the best balance between a small feature vector and high accuracy by exploring several feature sets. The results show that a small feature vector can achieve the same accuracy as a large feature vector if the extraction parameters are appropriately set, as seen in Figure 5.2. This finding could greatly benefit existing methods with high dimensionality and large memory footprints. It should be noted that, unlike many recent approaches, this study did not use any dimensionality reduction or subspace-learning methods and still achieved competitive results.

Table 5.16: Comparison with state-of-the-art methods

Database	Ref	Year	Features	Samples	Classifier	No. classes	Accuracy (Measure/Train-test)
CK+	[118]	2017	LTP + HOG	610	SVM	7	96% (10-fold)
CK+	[116]	2018	MRDTP + MRDNP	1281	ELM-RBF	7	98.4% (10-fold)
CK+	[119]	2017	Deep	927	FRR-CNN	6	92.06% (10-fold)
CK+	[111]	2017	ALDP	-	SVM	7	97% (80%-20%)
CK+	[32]	2017	Deep	927	FN2EN	6	98.6% (10-fold)
CK+	[31]	2019	HOG	634	MLP	8	82.97% (10-fold)
RFD	[117]	2014	PHOG	630	KNN	7	100% (67%-33%)
RFD	[120]	2012	PZM	-	KNN	6	94.51% (70%-30%)
RFD	[115]	2020	HOG + LLE	469	SVM	6	93.54% (10-fold)
CK+	Proposed	2022	HOG	347	MLP	6	98.4% (10-fold)
RFD	Proposed	2022	CLBP	402	SVM	6	97.5% (10-fold)

5.5 Chapter Summary

This study evaluated different combinations of local descriptors and classifiers for FER. The experiments comprehensively evaluated six descriptors and four classifiers on two famous datasets, such as CK+ and RFD. Notably, the appropriate choice of extraction parameters improved the considered descriptors' performances. The HOG descriptor achieved the most consistent performance in all the datasets. The computational costs of each classifier and descriptor were also studied, and the NB and MLP classifiers provided the fastest predictions. However, MLP can be time-consuming during training compared to the other classifiers. When evaluating the classifiers' recognition efficiencies, the results showed that SVM and MLP achieved the best recognition rates.

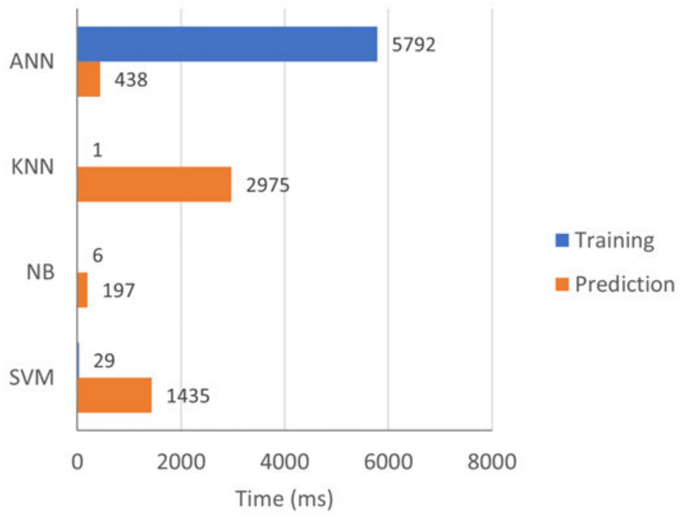


Figure 5.7: Prediction and training times of different classifiers.

In summary, of the six considered descriptors, HOG was one of the most promising, while SVM and MLP ranked at the top of the classifier choices. It was also found that SVM was faster to train and required little model tuning compared to MLP. On the other hand, MLP was notably faster at making predictions, which could be an advantage in real-time applications.

CHAPTER 6
CONCLUSIONS AND FUTURE WORK

This chapter presents the contributions of this research and proposes some directions for future work regarding the work presented in this dissertation.

6.1 Factors Affecting Facial Expression Recognition Systems

This section summarizes the findings concerning the factors affecting FER systems.

- Datasets containing incorrectly labeled images tend to introduce ambiguity between expressions, which brings down the accuracy of predictions. Hence, datasets must be carefully inspected (if possible, by at least two individuals) before being used.
- The registration step is also crucial to increasing the classification performance of a FER system. A correctly registered frontal face image must have all the facial components to guarantee the extraction of the best features and high prediction accuracy. Face registration also helps remove non-expression-specific pixels such as the neck, hair, and ears.
- Extraction parameters such as block size and the number of histogram bins directly impact the performance of histogram-based local descriptors. The experimental results show that as the sub-region sizes increase, the recognition rate improves and then depreciates, revealing that there exists an optimum parameter selection for any particular descriptor.
- Different classification techniques have different strengths and weaknesses, making them more useful in some contexts than others. MLP and SVM have the best clustering ability and work well with local descriptors. However, MLP is faster at making predictions than SVM.

6.2 Concluding Remarks

Research in Facial Expression Recognition has enabled the detection of human emotions for many artificial intelligence applications. This research was centered on developing FER systems using local descriptors and conventional classification methods. The proposed methodology for recognizing expressions consisted of three steps: i) face acquisition, ii) feature extraction and feature vector calculation, and iii) classification. The following is a summary of the findings of this research.

A comprehensive study was done on the HOG descriptor to understand and improve existing local descriptors. This study inspired this research to propose a new face registration algorithm. Furthermore, a new approach for optimized Facial Expression Recognition was proposed, which finds the parameter values for extracting local descriptors and normalizes them based on the sub-region size. Results showed that the performances of four local descriptors (i.e., HOG, LBP, CLBP, and WLD) were better than in previous studies. The system was also tested in real-time and recorded video data, achieving satisfactory performance. Furthermore, the experimental results revealed that face registration had a sizable impact on recognition performance.

The proposed FER approach was also implemented with different combinations of local descriptors and classification methods. The results showed that the HOG descriptor achieved the best performance across all the datasets. NB and MLP classifiers were the fastest at making predictions, while SVM and MLP classifiers produced the best recognition rates. Although MLP had a higher training cost, it was remarkably faster at making predictions than SVM. Hence, the choice between MLP and SVM appears to be a trade-off between training and prediction times.

6.3 Future Work

FER systems based on local descriptors and classification are simpler to develop and cost relatively less than more advanced solutions like deep learning. These advantages make local descriptors a competitive solution for mobile platforms and real-time applications with limited resources. However, there are some aspects of this research that, if improved, could benefit the overall performance of FER systems. Notable future work directions include:

- Analyze the effects of varying image conditions such as white noise, non-uniform illuminations, and different poses to assess the performance of FER methods in real-world situations. A dataset such as the BU-3DFE dataset [84] could be used to analyze pose variation, while the Affectnet dataset [83] contains real-world faces taken from the internet.

- Expression-specific features are the key to developing a robust facial expression descriptor. Local descriptors that utilize the gradient or edge information are more robust to image variants than intensity-based approaches. Moreover, selecting specific feature types, such as in [54], can reduce the dimensionality of descriptors, improve extraction speed, and achieve the same or better recognition efficiency.

REFERENCES

[1] A. Dzedzickis, A. Kaklauskas, and V. Bucinskas, “Human emotion recognition: Review of sensors and methods,” *Sensors*, vol. 20, no. 3, p. 592, 2020.

[2] J.-U. Garbas, T. Ruf, M. Unfried, and A. Dieckmann, “Towards robust real-time valence recognition from facial expressions for market research applications,” in *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*, IEEE, 2013, pp. 570–575.

[3] P. Leijdekkers, V. Gay, and F. Wong, “Capturemyemotion: A mobile app to improve emotion learning for autistic children using sensors,” in *Proceedings of the 26th IEEE International Symposium on Computer-Based Medical Systems*, IEEE, 2013, pp. 381–384.

[4] N. Khediri, M. B. Ammar, and M. Kherallah, “Towards an online emotional recognition system for intelligent tutoring environment,” in *ACIT’2017 The International Arab Conference on Information Technology Yassmine Hammamet*, 2017.

[5] M. Feidakis, T. Daradoumis, and S. Caballé, “Endowing e-learning systems with emotion awareness,” in *2011 Third international conference on intelligent networking and collaborative systems*, IEEE, 2011, pp. 68–75.

[6] J. A. Russell, “A circumplex model of affect.,” *Journal of personality and social psychology*, vol. 39, no. 6, p. 1161, 1980.

[7] M. H. Kabir, T. Jabid, and O. Chae, “Local directional pattern variance (ldpv): A robust feature descriptor for facial expression recognition.,” *Int. Arab J. Inf. Technol.*, vol. 9, no. 4, pp. 382–391, 2012.

[8] H. Hassan and S. A. Suandi, “Analysis of local binary pattern for facial expression recognition using patch local binary pattern on extended cohn kanade database,” in *10th International Conference on Robotics, Vision, Signal Processing and Power Applications*, Springer, 2019, pp. 633–639.

[9] R. C. Rahul and M. Cherian, “Facial expression recognition using pca and texture-based ldn descriptor,” in *Proceedings of the International Conference on Soft Computing Systems*, Springer, 2016, pp. 113–122.

[10] W. Mellouk and W. Handouzi, “Facial emotion recognition using deep learning: Review and insights,” *Procedia Computer Science*, vol. 175, pp. 689–694, 2020.

[11] C. Turan and K.-M. Lam, “Histogram-based local descriptors for facial expression recognition (fer): A comprehensive study,” *Journal of visual communication and image representation*, vol. 55, pp. 331–341, 2018.

[12] P. Carcagnì, M. Del Coco, M. Leo, and C. Distante, “Facial expression recognition and histograms of oriented gradients: A comprehensive study,” *SpringerPlus*, vol. 4, no. 1, pp. 1–25, 2015.

[13] C. Shan, S. Gong, and P. W. McOwan, “Facial expression recognition based on local binary patterns: A comprehensive study,” *Image and vision Computing*, vol. 27, no. 6, pp. 803–816, 2009.

[14] M. Jonnalagedda and D. Doye, “Augmented local binary patterns for facial expression recognition,” *International Journal of Mathematics and Computation, CESER publications*, vol. volume 27, Page 59–68, Jan. 2016.

[15] K. Slimani, M. Kas, Y. El Merabet, Y. Ruichek, and R. Messoussi, “Local feature extraction based facial emotion recognition: A survey,” *International Journal of Electrical and Computer Engineering*, vol. 10, no. 4, p. 4080, 2020.

[16] M.-H. Yang, D. J. Kriegman, and N. Ahuja, “Detecting faces in images: A survey,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 1, pp. 34–58, 2002.

[17] P. Viola and M. J. Jones, “Robust real-time face detection,” *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[18] M. Mandal, M. Verma, S. Mathur, S. K. Vipparthi, S. Murala, and D. K. Kumar, “Regional adaptive affinitive patterns (radap) with logical operators for facial expression recognition,” *IET Image Processing*, vol. 13, no. 5, pp. 850–861, 2019.

[19] Z. Zhang, G. Lu, J. Yan, H. Li, N. Sun, and X. Li, “Compact local gabor directional number pattern for facial expression recognition,” *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 26, no. 3, pp. 1236–1248, 2018.

[20] B. T. Lau, “Portable real time emotion detection system for the disabled,” *Expert Systems with Applications*, vol. 37, no. 9, pp. 6561–6566, 2010.

[21] R. A. Patil, V. Sahula, and A. S. Mandal, “Facial expression recognition in image sequences using active shape model and svm,” in *2011 UKSIM 5th European Symposium on Computer Modeling and Simulation*, IEEE, 2011, pp. 168–173.

[22] B. Niu, Z. Gao, and B. Guo, “Facial expression recognition with lbp and orb features,” *Computational Intelligence and Neuroscience*, vol. 2021, 2021.

[23] D. E. King, “Dlib-ml: A machine learning toolkit,” *The Journal of Machine Learning Research*, vol. 10, pp. 1755–1758, 2009.

[24] B. C. Ko, “A brief review of facial emotion recognition based on visual information,” *sensors*, vol. 18, no. 2, p. 401, 2018.

[25] M. Pantic and L. J. Rothkrantz, “Facial action recognition for facial expression analysis from static face images,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 3, pp. 1449–1461, 2004.

[26] C. Loconsole, C. R. Miranda, G. Augusto, A. Frisoli, and V. Orvalho, “Real-time emotion recognition novel method for geometrical facial features extraction,” in *2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, IEEE, vol. 1, 2014, pp. 378–385.

[27] M. Gaeta, G. Iovane, and E. Sangineto, “A 3d geometric approach to face detection and facial expression recognition,” *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 9, no. 1, pp. 39–53, 2006.

[28] A. Uçar, Y. Demir, and C. Güzeliş, “A new facial expression recognition based on curvelet transform and online sequential extreme learning machine initialized with spherical clustering,” *Neural Computing and Applications*, vol. 27, no. 1, pp. 131–142, 2016.

[29] H. Mahersia and K. Hamrouni, “Using multiple steerable filters and bayesian regularization for facial expression recognition,” *Engineering Applications of Artificial Intelligence*, vol. 38, pp. 190–202, 2015.

[30] A. Poursaberi, H. A. Noubari, M. Gavrilova, and S. N. Yanushkevich, “Gauss–laguerre wavelet textural feature fusion with geometrical information for facial expression identification,” *EURASIP Journal on Image and Video Processing*, vol. 2012, no. 1, pp. 1–13, 2012.

[31] H. I. Dino and M. B. Abdulrazzaq, “Facial expression classification based on svm, knn and mlp classifiers,” in *2019 International Conference on Advanced Science and Engineering (ICOASE)*, IEEE, 2019, pp. 70–75.

[32] H. Ding, S. K. Zhou, and R. Chellappa, “Facenet2expnet: Regularizing a deep face recognition net for expression recognition,” in *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, IEEE, 2017, pp. 118–126.

[33] S. Zhang, X. Pan, Y. Cui, X. Zhao, and L. Liu, “Learning affective video features for facial expression recognition via hybrid deep learning,” *IEEE Access*, vol. 7, pp. 32 297–32 304, 2019.

[34] F. Bougourzi, F. Dornaika, K. Mokrani, A. Taleb-Ahmed, and Y. Ruichek, “Fusing transformed deep and shallow features (ftds) for image-based facial expression recognition,” *Expert Systems with Applications*, vol. 156, p. 113 459, 2020.

[35] C. Dong, R. Wang, and Y. Hang, “Facial expression recognition based on improved vgg convolutional neural network,” in *Journal of Physics: Conference Series*, IOP Publishing, vol. 2083, 2021, p. 032 030.

[36] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, and T. Chen, “Recent advances in convolutional neural networks,” *Pattern recognition*, vol. 77, pp. 354–377, 2018.

[37] T. Ojala, M. Pietikäinen, and D. Harwood, “A comparative study of texture measures with classification based on featured distributions,” *Pattern recognition*, vol. 29, no. 1, pp. 51–59, 1996.

[38] G. Benitez-Garcia, T. Nakamura, and M. Kaneko, “Facial expression recognition based on local fourier coefficients and facial fourier descriptors,” *Journal of Signal and Information Processing*, vol. 8, no. 3, pp. 132–151, 2017.

[39] V. HJ and S. Math, “Face and facial expression recognition using local directional feature structure,” *International Journal of Nonlinear Analysis and Applications*, vol. 13, no. 1, pp. 1067–1079, 2022.

[40] T. Mohammad and M. L. Ali, “Robust facial expression recognition based on local monotonic pattern (lmp),” in *14th International Conference on Computer and Information Technology (ICCIT 2011)*, IEEE, 2011, pp. 572–576.

[41] F. Bashar, A. Khan, F. Ahmed, and M. H. Kabir, “Robust facial expression recognition based on median ternary pattern (mtp),” in *2013 International Conference on Electrical Information and Communication Technology (EICT)*, IEEE, 2014, pp. 1–5.

[42] M. S. Islam and S Auwatanamo, “Facial expression recognition using local arc pattern,” *Trends Appl. Sci. Res*, vol. 9, no. 2, p. 113, 2014.

[43] T. Jabid and O. Chae, “Facial expression recognition based on local transitional pattern,” *International Information Institute (Tokyo). Information*, vol. 15, no. 5, p. 2007, 2012.

[44] X. Fu and W. Wei, “Centralized binary patterns embedded with image euclidean distance for facial expression recognition,” in *2008 Fourth International Conference on Natural Computation*, IEEE, vol. 4, 2008, pp. 115–119.

[45] M. S. Islam, “Local gradient pattern-a novel feature representation for facial expression recognition,” *Journal of AI and Data Mining*, vol. 2, no. 1, pp. 33–38, 2014.

[46] B. Santra and D. P. Mukherjee, “Local dominant binary patterns for recognition of multi-view facial expressions,” in *Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing*, 2016, pp. 1–8.

[47] F. Makhmudkhujaev, M. Abdullah-Al-Wadud, M. T. B. Iqbal, B. Ryu, and O. Chae, “Facial expression recognition with local prominent directional pattern,” *Signal Processing: Image Communication*, vol. 74, pp. 1–12, 2019.

[48] F Ahmed, “Gradient directional pattern: A robust feature descriptor for facial expression recognition,” *Electronics letters*, vol. 48, no. 19, pp. 1203–1204, 2012.

[49] F. Ahmed and E. Hossain, “Automated facial expression recognition using gradient-based ternary texture patterns,” *Chin. J. Eng*, vol. 2013, p. 831 747, 2013.

[50] A. R. Rivera, J. R. Castillo, and O. O. Chae, "Local directional number pattern for face analysis: Face and expression recognition," *IEEE transactions on image processing*, vol. 22, no. 5, pp. 1740–1752, 2012.

[51] L. Zhou and H. Wang, "Local gradient increasing pattern for facial expression recognition," in *2012 19th IEEE International Conference on Image Processing*, IEEE, 2012, pp. 2601–2604.

[52] M. S. Islam, "Local gray code pattern (lgcp): A robust feature descriptor for facial expression recognition," *International Journal of Science and Research (IJSR)*, vol. 2, no. 8, pp. 413–419, 2013.

[53] T. Jabid, M. H. Kabir, and O. Chae, "Local directional pattern (ldp) for face recognition," in *2010 digest of technical papers international conference on consumer electronics (ICCE)*, IEEE, 2010, pp. 329–330.

[54] S. Shojaeilangari, W.-Y. Yau, J. Li, and E.-K. Teoh, "Feature extraction through binary pattern of phase congruency for facial expression recognition," in *2012 12th International Conference on Control Automation Robotics & Vision (ICARCV)*, IEEE, 2012, pp. 166–170.

[55] T. Ahsan, T. Jabid, and U.-P. Chong, "Facial expression recognition using local transitional pattern on gabor filtered facial images," *IETE Technical Review*, vol. 30, Jan. 2013. DOI: `10.4103/0256-4602.107339`.

[56] S. Z. Ishraque, A. H. Banna, and O. Chae, "Local gabor directional pattern for facial expression recognition," in *2012 15th International Conference on Computer and Information Technology (ICCIT)*, IEEE, 2012, pp. 164–167.

[57] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[58] P. Kovesi, "Image features from phase congruency," 1995.

[59] C. Liu, K. Hirota, J. Ma, Z. Jia, and Y. Dai, "Facial expression recognition using hybrid features of pixel and geometry," *Ieee Access*, vol. 9, pp. 18 876–18 889, 2021.

[60] A. Burkov, *The hundred-page machine learning book*. Andriy Burkov Quebec City, QC, Canada, 2019, vol. 1.

[61] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise.," in *kdd*, vol. 96, 1996, pp. 226–231.

[62] A. A. Razzak and A. R. Hashem, "Facial expression recognition using hybrid transform," *International Journal of Computer Applications*, vol. 119, no. 15, 2015.

[63] S. Zhang, X. Zhao, and B. Lei, "Facial expression recognition based on local binary patterns and local fisher discriminant analysis," *WSEAS transactions on signal processing*, vol. 8, no. 1, pp. 21–31, 2012.

[64] G. Muhammad, M. Alsulaiman, S. U. Amin, A. Ghoneim, and M. F. Alhamid, “A facial-expression monitoring system for improved healthcare in smart cities,” *IEEE Access*, vol. 5, pp. 10 871–10 881, 2017.

[65] L. T. Dang, E. W. Cooper, and K. Kamei, “Development of facial expression recognition for training video customer service representatives,” in *2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, IEEE, 2014, pp. 1297–1303.

[66] K. Takahashi and Y. Mitsukura, “An entertainment robot based on head pose estimation and facial expression recognition,” in *2012 Proceedings of SICE Annual Conference (SICE)*, IEEE, 2012, pp. 2057–2061.

[67] I. M. Revina and W. S. Emmanuel, “A survey on human face expression recognition techniques,” *Journal of King Saud University-Computer and Information Sciences*, vol. 33, no. 6, pp. 619–628, 2021.

[68] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[69] D. Ghimire, S. Jeong, J. Lee, and S. H. Park, “Facial expression recognition based on local region specific features and support vector machines,” *Multimedia Tools and Applications*, vol. 76, no. 6, pp. 7803–7821, 2017.

[70] A. Saeed, A. Al-Hamadi, R. Niese, and M. Elzobi, “Frame-based facial expression recognition using geometrical features,” *Advances in human-computer interaction*, vol. 2014, 2014.

[71] N. Sebe, M. S. Lew, I. Cohen, A. Garg, and T. S. Huang, “Emotion recognition using a cauchy naive bayes classifier,” in *Object recognition supported by user interaction for service robots*, IEEE, vol. 1, 2002, pp. 17–20.

[72] S. Rajan, P. Chenniappan, S. Devaraj, and N. Madian, “Facial expression recognition techniques: A comprehensive survey,” *IET Image Processing*, vol. 13, no. 7, pp. 1031–1040, 2019.

[73] M. Schmidt, M. Schels, and F. Schwenker, “A hidden markov model based approach for facial expression recognition in image sequences,” in *IAPR workshop on artificial neural networks in pattern recognition*, Springer, 2010, pp. 149–160.

[74] A. Kamila, “Evaluation of selected data mining algorithms implemented in Medical Decision Support Systems,” M.S. thesis, School of Engineering, Ronneby, Sweden, 200.

[75] X. Li, G. C. Nsofor, and L. Song, “A comparative analysis of predictive data mining techniques,” *International Journal of Rapid Manufacturing*, vol. 1, no. 2, pp. 150–172, 2009.

[76] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: Synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[77] A. Mishra, *Metrics to evaluate your machine learning algorithm*, [Accessed 31-January-2022], 2022. [Online]. Available: `https://en.wikipedia.org/w/index.php?title=English_Wikipedia&oldid=1104796737`.

[78] K. J. Danjuma, "Performance evaluation of machine learning algorithms in post-operative life expectancy in the lung cancer patients," *arXiv preprint arXiv:1504.04646*, 2015.

[79] I. H. Witten and E. Frank, "Data mining: Practical machine learning tools and techniques, second edition," pp. 168–169, 2005.

[80] R. L. Kristensen, Z.-H. Tan, Z. Ma, and J. Guo, "Binary pattern flavored feature extractors for facial expression recognition: An overview," in *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, IEEE, 2015, pp. 1131–1137.

[81] T. Kanade, J. F. Cohn, and Y. Tian, "Comprehensive database for facial expression analysis," in *Proceedings fourth IEEE international conference on automatic face and gesture recognition (cat. No. PR00580)*, IEEE, 2000, pp. 46–53.

[82] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews, "The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression," in *2010 ieee computer society conference on computer vision and pattern recognition-workshops*, IEEE, 2010, pp. 94–101.

[83] A. Mollahosseini, B. Hasani, and M. H. Mahoor, "Affectnet: A database for facial expression, valence, and arousal computing in the wild," *IEEE Transactions on Affective Computing*, vol. 10, no. 1, pp. 18–31, 2017.

[84] C. Eroglu Erdem, C. Turan, and Z. Aydin, "Baum-2: A multilingual audio-visual affective face database," *Multimedia tools and applications*, vol. 74, no. 18, pp. 7429–7459, 2015.

[85] M. J. Lyons, S. Akamatsu, M. Kamachi, J. Gyoba, and J. Budynek, "The japanese female facial expression (jaffe) database," in *Proceedings of third international conference on automatic face and gesture recognition*, 1998, pp. 14–16.

[86] N. Aifanti, C. Papachristou, and A. Delopoulos, "The mug facial expression database," in *11th International Workshop on Image Analysis for Multimedia Interactive Services WIAMIS 10*, IEEE, 2010, pp. 1–4.

[87] P. Lucey, J. F. Cohn, K. M. Prkachin, P. E. Solomon, S. Chew, and I. Matthews, "Painful monitoring: Automatic pain monitoring using the unbc-mcmaster shoulder pain expression archive database," *Image and Vision Computing*, vol. 30, no. 3, pp. 197–205, 2012.

[88] O. Langner, R. Dotsch, G. Bijlstra, D. H. Wigboldus, S. T. Hawk, and A. Van Knippenberg, "Presentation and validation of the radboud faces database," *Cognition and emotion*, vol. 24, no. 8, pp. 1377–1388, 2010.

[89] S. Happy and A. Routray, "Automatic facial expression recognition using features of salient facial patches," *IEEE transactions on Affective Computing*, vol. 6, no. 1, pp. 1–12, 2014.

[90] X. Zhao and S. Zhang, "Facial expression recognition based on local binary patterns and kernel discriminant isomap," *Sensors*, vol. 11, no. 10, pp. 9573–9588, 2011.

[91] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[92] R. C. Gonzalez and R. E. Woods, *Digital image processing: Pearson international edition*, 2008.

[93] F. Bolelli, S. Allegretti, L. Baraldi, and C. Grana, "Spaghetti labeling: Directed acyclic graphs for block-based connected components labeling," *IEEE Transactions on Image Processing*, vol. 29, pp. 1999–2012, 2019.

[94] J. J. Lim, C. L. Zitnick, and P. Dollár, "Sketch tokens: A learned mid-level representation for contour and object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 3158–3165.

[95] J. Sklansky, "Finding the convex hull of a simple polygon," *Pattern Recognition Letters*, vol. 1, no. 2, pp. 79–83, 1982.

[96] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, Ieee, vol. 1, 2005, pp. 886–893.

[97] J. Chen, Z. Chen, Z. Chi, and H. Fu, "Facial expression recognition based on facial components detection and hog features," in *International workshops on electrical and computer engineering subfields*, 2014, pp. 884–888.

[98] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, *A practical guide to support vector classification*, 2003.

[99] B. Da and N. Sang, "Local binary pattern based face recognition by estimation of facial distinctive information distribution," *Optical Engineering*, vol. 48, no. 11, p. 117 203, 2009.

[100] F. Ahmed, E. Hossain, A. H. Bari, and A. Shihavuddin, "Compound local binary pattern (clbp) for robust facial expression recognition," in *2011 IEEE 12th International Symposium on Computational Intelligence and Informatics (CINTI)*, IEEE, 2011, pp. 391–395.

[101] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face recognition with local binary patterns," in *European conference on computer vision*, Springer, 2004, pp. 469–481.

[102] J. Chen, S. Shan, C. He, G. Zhao, M. Pietikäinen, X. Chen, and W. Gao, "Wld: A robust local image descriptor," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1705–1720, 2009.

[103] Q. Jia, X. Gao, H. Guo, Z. Luo, and Y. Wang, "Multi-layer sparse representation for weighted lbp-patches based facial expression recognition," *Sensors*, vol. 15, no. 3, pp. 6719–6739, 2015.

[104] N. Hesse, T. Gehrig, H. Gao, and H. K. Ekenel, "Multi-view facial expression recognition using local appearance features," in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, IEEE, 2012, pp. 3533–3536.

[105] X. Lu, L. Kong, M. Liu, and X. Zhang, "Facial expression recognition based on gabor feature and src," in *Chinese Conference on Biometric Recognition*, Springer, 2015, pp. 416–422.

[106] M. Sajjad, A. Shah, Z. Jan, S. I. Shah, S. W. Baik, and I. Mehmood, "Facial appearance and texture feature-based robust facial expression recognition framework for sentiment knowledge discovery," *Cluster Computing*, vol. 21, no. 1, pp. 549–567, 2018.

[107] H. P. Truong, T. P. Nguyen, and Y.-G. Kim, "Weighted statistical binary patterns for facial feature representation," *Applied Intelligence*, pp. 1–20, 2021.

[108] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 915–928, 2007.

[109] C. Guo, J. Liang, G. Zhan, Z. Liu, M. Pietikäinen, and L. Liu, "Extended local binary patterns for efficient and robust spontaneous facial micro-expression recognition," *IEEE Access*, vol. 7, pp. 174 517–174 530, 2019.

[110] D. G. R. Kola and S. K. Samayamantula, "A novel approach for facial expression recognition using local binary pattern with adaptive window," *Multimedia Tools and Applications*, vol. 80, no. 2, pp. 2243–2262, 2021.

[111] A. M. Shabat and J.-R. Tapamo, "Angled local directional pattern for texture analysis with an application to facial expression recognition," *IET Computer Vision*, vol. 12, no. 5, pp. 603–608, 2018.

[112] A. S. M. Sohail and P. Bhattacharya, "Classification of facial expressions using k-nearest neighbor classifier," in *International Conference on Computer Vision/Computer Graphics Collaboration Techniques and Applications*, Springer, 2007, pp. 555–566.

[113] G. Panchal and K Pushpalatha, "A local binary pattern based facial expression recognition using k-nearest neighbor (knn) search," *Int. J. Eng. Res. Technol.*, vol. 6, no. 5, pp. 525–530, 2017.

[114] H. Boughrara, M. Chtourou, C. Ben Amar, and L. Chen, "Facial expression recognition based on a mlp neural network using constructive training algorithm," *Multimedia Tools and Applications*, vol. 75, no. 2, pp. 709–731, 2016.

[115] Y. Yaddaden, M. Adda, and A. Bouzouane, "Facial expression recognition using locally linear embedding with lbp and hog descriptors," in *2020 2nd International Workshop on Human-Centric Smart Environments for Health and Well-being (IHSH)*, IEEE, 2021, pp. 221–226.

[116] A. S. Alphonse and D. Dharma, “Novel directional patterns and a generalized supervised dimension reduction system (gsdrs) for facial emotion recognition,” *Multimedia Tools and Applications*, vol. 77, no. 8, pp. 9455–9488, 2018.

[117] S. Shokrani, P. Moallem, and M. Habibi, “Facial emotion recognition method based on pyramid histogram of oriented gradient over three direction of head,” in *2014 4th International Conference on Computer and Knowledge Engineering (ICCKE)*, IEEE, 2014, pp. 215–220.

[118] K. Lekdioui, R. Messoussi, Y. Ruichek, Y. Chaabi, and R. Touahni, “Facial decomposition for expression recognition using texture/shape descriptors and svm classifier,” *Signal Processing: Image Communication*, vol. 58, pp. 300–312, 2017.

[119] S. Xie and H. Hu, “Facial expression recognition with frr-cnn,” *Electronics Letters*, vol. 53, no. 4, pp. 235–237, 2017.

[120] H. R. Kanan and M. Ahmady, “Recognition of facial expressions using locally weighted and adjusted order pseudo zernike moments,” in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, IEEE, 2012, pp. 3419–3422.