

Electronic Theses and Dissertations, 2020-

2020

Increasing Accuracy Performance through Optimal Feature Extraction Algorithms

Genevieve Sapijaszko
University of Central Florida

 Part of the [Electrical and Computer Engineering Commons](#)
Find similar works at: <https://stars.library.ucf.edu/etd2020>
University of Central Florida Libraries <http://library.ucf.edu>

This Doctoral Dissertation (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Electronic Theses and Dissertations, 2020- by an authorized administrator of STARS. For more information, please contact STARS@ucf.edu.

STARS Citation

Sapijaszko, Genevieve, "Increasing Accuracy Performance through Optimal Feature Extraction Algorithms" (2020). *Electronic Theses and Dissertations, 2020-*. 129.
<https://stars.library.ucf.edu/etd2020/129>



INCREASING ACCURACY PERFORMANCE THROUGH OPTIMAL FEATURE
EXTRACTION ALGORITHMS

by

GENEVIÈVE MARIE ISABELLE SAPIJASZKO
B.Sc.E.E. University of Calgary, 1994
M.S.E.E. University of Calgary, 2000

A dissertation submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy
in the Department of Electrical and Computer Engineering
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Spring Term
2020

Major Professor: Wasfy B. Mikhael

© 2020 GENEVIÈVE MARIE ISABELLE SAPIJASZKO

ABSTRACT

This research developed models and techniques to improve the three key modules of popular recognition systems: preprocessing, feature extraction, and classification. Improvements were made in four key areas: processing speed, algorithm complexity, storage space, and accuracy. The focus was on the application areas of the face, traffic sign, and speaker recognition. In the preprocessing module of facial and traffic sign recognition, improvements were made through the utilization of grayscaling and anisotropic diffusion. In the feature extraction module, improvements were made in two different ways; first, through the use of mixed transforms and second through a convolutional neural network (CNN) that best fits specific datasets. The mixed transform system consists of various combinations of the Discrete Wavelet Transform (DWT) and Discrete Cosine Transform (DCT), which have a reliable track record for image feature extraction. In terms of the proposed CNN, a neuroevolution system was used to determine the characteristics and layout of a CNN to best extract image features for particular datasets. In the speaker recognition system, the improvement to the feature extraction module comprised of a quantized spectral covariance matrix and a two-dimensional Principal Component Analysis (2DPCA) function. In the classification module, enhancements were made in visual recognition through the use of two neural networks: the multi-layer sigmoid and convolutional neural network. Results show that the proposed improvements in the three modules led to an increase in accuracy as well as reduced algorithmic complexity, with corresponding reductions in storage space and processing time.

This dissertation is dedicated to my daughter, Jamie, my sister, Julie, and my mom, Marie, for their prayers and all-around support in every area. I would also like to acknowledge my friends; you made sure I had the moral and emotional support I needed. When I needed to chat, find encouragement, you took time out of your busy day to give me supporting words and strength.

Your help can never be understated.

ACKNOWLEDGMENTS

First and foremost, I would like to thank God, whose many blessings have made me who I am today. The following document summarizes many years worth of effort, frustration, and achievement. I am forever grateful to Dr. Wasfy Mikhael for his patience, guidance, charisma, and effort, for they were essential to the creation of this document.

TABLE OF CONTENTS

LIST OF FIGURES	xvi
LIST OF TABLES	xix
DEFINITION OF KEY TERMS	xxv
CHAPTER 1: INTRODUCTION	1
1.1 Visual Recognition	2
1.1.1 Applications of Face Recognition	3
1.1.2 Face Recognition Advantages	3
1.1.3 Issues and Concerns with Face Recognition	4
1.1.4 Applications of Traffic Sign Recognition	4
1.1.5 Issues and Concerns with Traffic Sign Recognition	5
1.1.6 Issues and Concerns with Over-Designed Networks for Image Recognition	5
1.2 Speaker Recognition	6
1.2.1 Applications of Speaker Recognition	7
1.2.2 Issues and Concerns with Speaker Recognition	8

1.3	Datasets	8
1.3.1	Face Datasets	8
1.3.1.1	Olivetti Research Lab (ORL) Database	9
1.3.1.2	YALE	10
1.3.1.3	The Facial Recognition Technology (FERET-fc)	10
1.3.1.4	FEI Dataset	11
1.3.2	Object Datasets	12
1.3.2.1	Tiny Imagenet	12
1.3.3	Traffic Signs Datasets	13
1.3.3.1	BelgiumTS - Belgian Traffic Sign Dataset (BTS)	13
1.3.3.2	GTSRB	14
1.3.3.3	TSRD	15
1.3.4	Handwritten Digits Dataset	16
1.3.4.1	MNIST	16
1.3.5	Speaker Recognition Datasets	17
1.3.5.1	Spoken Language Understanding (CLSU) Speaker Recognition Corpus Data Collection (Version 1.1)	17
1.3.5.2	TIMIT	18

1.3.5.3	NOIZEOUS	19
1.4	Organization of the Dissertation	20
CHAPTER 2: LITERATURE REVIEW		22
2.1	Introduction	22
2.2	Transform Domain	22
2.3	Neural Network Approach	24
2.3.1	Multilayer Sigmoid Neural Network	24
2.3.2	Convolutional Neural Network (CNN)	26
2.3.3	Neuroevolution	28
2.4	Unsupervised Learning Algorithms	30
2.4.1	Vector Quantization	30
2.4.2	Linear Transformation	31
2.5	Enhancing the Performance of a Facial Recognition System	31
CHAPTER 3: METHODOLOGY		33
3.1	Preprocessing	33
3.1.1	Grayscaleing	33
3.1.2	Image Resize	34

3.1.3	Anisotropic Diffusion	34
3.2	Image Feature Extraction	35
3.2.1	Two-Dimensional Discrete Wavelet Transform (2D-DWT)	36
3.2.2	Two-Dimensional Discrete Cosine Transform (2D-DCT)	39
3.2.2.1	Biorthogonal Wavelets	40
3.2.3	Neuroevolution	41
3.2.4	Convolutional Neural Networks	42
3.2.4.1	Pooling layer	44
3.2.4.2	Fully Connected layer	45
3.2.4.3	Forward and Backward Propagation	45
3.2.4.4	Activation Functions	48
3.3	Image Classification	57
3.3.1	Multilayer Perceptron Neural Network (MLPNN)	57
3.3.2	Multilayer Sigmoid Neural Network (MLSNN) Classifier	59
3.3.3	Pretrained Convolutional Neural Networks	60
3.3.3.1	AlexNet	61
3.3.3.2	VGGNet	61

3.3.3.3	GoogLeNet	62
3.3.3.4	Inception-v3	63
3.3.3.5	ResNet	64
3.4	Window Based Feature Extraction Algorithms	65
3.4.1	Real Cepstral Coefficients (RCC)	65
3.4.2	Mel Frequency Cepstral Coefficients (MFCC)	65
3.4.3	Delta-Mel Frequency Cepstral Coefficients	66
3.4.4	Linear Prediction Coefficients (LPC)	66
3.4.5	Linear Prediction Cepstral Coefficients (LPCC)	67
3.4.6	Perceptual Linear Predictive Cepstral Coefficients (PLPCC)	67
3.4.7	RelAtive SpecTrAl (Rasta) - PLPCC	68
CHAPTER 4: FACIAL RECOGNITION SYSTEM USING MIXED TRANSFORM AND MULTILAYER SIGMOID NEURAL NETWORK CLASSIFIER		69
4.1	Introduction	69
4.2	Proposed System	73
4.2.1	Feature Extraction	73
4.2.2	Classification	75

4.3	Experimental Results	79
4.3.1	Remarks on the Results	86
4.4	Conclusion	89
CHAPTER 5: DESIGNING CONVOLUTIONAL NEURAL NETWORKS FOR VARIOUS IMAGE RECOGNITION USING NEUROEVOLUTION		90
5.1	Introduction	90
5.1.1	Convolutional Neural Networks (CNN)	90
5.1.2	Feature Extraction	91
5.1.3	Neuroevolution (NE)	93
5.2	Proposed System	95
5.2.1	Preprocessing	95
5.2.2	Convolutional Neural Network (CNN)	95
5.2.2.1	Aspect of Optimization	96
5.2.3	Neuroevolution (NE)	100
5.2.3.1	Initial Population	102
5.2.3.2	Fitness Function	102
5.2.3.3	Selection	102

5.2.3.4	Crossover	103
5.2.3.5	Mutation	103
5.2.3.6	Termination	103
5.2.3.7	Comments	104
5.3	Experimental Results	104
5.3.1	Remarks on the Results	110
5.4	Conclusion	111
CHAPTER 6: ADAPTIVE FEATURE EXTRACTION ALGORITHM USING MIXED TRANS-		
FORMS FOR FACIAL RECOGNITION		113
6.1	Introduction	113
6.2	Proposed system	114
6.3	Experimental Results	118
6.3.1	System Performance Evaluation using the ORL Database	120
6.3.2	System Performance Evaluation using the YALE Dataset	121
6.3.3	System Performance Evaluation using the FERET-fc Dataset	122
6.3.4	Remarks on the Performance of the Proposed System	122
6.4	Conclusions	122

CHAPTER 7: AN OVERVIEW OF RECENT CONVOLUTIONAL NEURAL NETWORK ALGORITHMS FOR IMAGE RECOGNITION	124
7.1 Introduction	124
7.2 Proposed System	125
7.3 Experimental Results	127
7.4 Conclusions	132
CHAPTER 8: TRAFFIC SIGN RECOGNITION BASED ON MULTILAYER PERCEP- TRON USING DWT AND DCT	133
8.1 Introduction	133
8.2 Proposed System	134
8.3 Experimental Results	137
8.3.1 System Performance Evaluation using the BTS, TSRD and GTSRB Dataset	139
8.3.2 Remarks on the Performance of the Proposed Systems	139
8.4 Conclusions	140
CHAPTER 9: ROBUST SPEAKER RECOGNITION SYSTEM EMPLOYING COVARI- ANCE MATRIX AND EIGENVOICE	141
9.1 Introduction	141
9.2 Proposed System	142

9.2.1	Spectral Quantization	144
9.2.2	Eigenvoice	149
9.2.3	Registered User Database of Speaker Features	149
9.2.4	Classification	150
9.3	Experimental Results	150
9.4	Conclusions	152
CHAPTER 10: AN OVERVIEW OF RECENT WINDOW BASED FEATURE EXTRACTION ALGORITHMS FOR SPEAKER RECOGNITION		153
10.1	Introduction	153
10.2	Proposed System	154
10.3	Experimental Results	156
10.4	Conclusions	158
CHAPTER 11: CONCLUSION AND FUTURE WORK		160
11.1	Future Work	161
APPENDIX A: PERMISSION LETTERS TO REPRINT THE ARTICLES IN THIS DISSERTATION		162
APPENDIX B: SAMPLE SIZE DETERMINATION SOFTWARE CODE		166

LIST OF REFERENCES	177
------------------------------	-----

LIST OF FIGURES

Figure 1.1: Image Analysis Steps	2
Figure 1.2: Speech Processing	7
Figure 1.3: Sample Images from the ORL Dataset	9
Figure 1.4: Sample Images from the YALE Dataset	10
Figure 1.5: Sample Images from the FERET-fc Dataset	11
Figure 1.6: Sample Images for FEI Dataset	12
Figure 1.7: Sample Images from the BTS Dataset	14
Figure 1.8: Sample Images from the GTSRB Dataset	15
Figure 1.9: Sample Images for TSRD Dataset	16
Figure 1.10 Sample Images for MNIST Dataset	17
Figure 3.1: Block Diagram of Discrete Wavelet Filter Analysis	38
Figure 3.2: Illustration of the Typical Generational Neuroevolution Process	41
Figure 3.3: Basic Convolutional Neural Network	43
Figure 3.4: Model of Neural Network	49
Figure 3.5: Comparison of Activation Functions	52

Figure 3.6: Comparison of Derivative of Activation Functions	52
Figure 3.7: Model of Convolutional Neural Network Showing Parameters	55
Figure 4.1: Block Diagram of Proposed System	73
Figure 4.2: Block Diagram of Proposed Feature Extraction System	74
Figure 4.3: Proposed System with MLSNN	79
Figure 4.4: Training Stage of Systems 1 through 6.	80
Figure 4.5: Recognition Accuracy for FEI Dataset Across Different DCT Compressed Sizes During Training of the System	82
Figure 4.6: Recognition Accuracy and Cross-Entropy Loss of Combined Dataset for the Proposed System	88
Figure 5.1: Details of Chromosome in the GA for all Datasets	104
Figure 5.2: Recognition Accuracy for ALL Datasets	106
Figure 6.1: <i>Block Diagram of Proposed Facial Identification System</i>	115
Figure 8.1: Proposed Traffic Sign Recognition System	137
Figure 9.1: Block Diagram of MFCC Feature Extraction Model	143
Figure 9.2: Speaker Recognition System	148
Figure 9.3: Average Recognition Rate of the Five Datasets Using Various Matrices	151

Figure 9.4: Recognition Rate of the Five Datasets under Train Noise for Various SNR . .	151
Figure 10.1 Experimental Setup of Speaker Identification System	155
Figure 10.2 Frame Length and Overlap	156
Figure 10.3 Recognition Rate of Feature Extraction Methods	157
Figure 10.4 Feature Extraction Methods	158
Figure A.1: Reprint permission request for the Recent Convolutional Neural Networks Article	163
Figure A.2: Reprint permission request for the Traffic Sign Recognition System Article .	164
Figure A.3: Reprint permission request for the Adaptive Feature Extraction Algorithm Article	164
Figure A.4: Reprint permission request for the Overview of Recent Window Based Sys- tems Article	165
Figure A.5: Reprint permission request for the Robust Speaker Recognition Article	165

LIST OF TABLES

Table 1.1: Image Analysis Categories [1]	3
Table 1.2: CLSU Speech Data Utterances	18
Table 1.3: Timit Dataset Speech Phonetic label	19
Table 4.1: Image Dimension Comparison for the ORL, YALE, FERET-c, FEI and Com- bined Dataset and Proposed System for Classification	83
Table 4.2: Recognition Accuracy (%) For All Datasets	84
Table 4.3: Training + Testing Processing Time (s) Per Image for All Datasets	85
Table 4.4: Testing Processing Time (s) per Image for ALL Datasets	86
Table 5.1: Image Dimension Comparison for the ORL, YALE, FERET-fc, FEI and Com- bined Dataset and Proposed System for Classification	107
Table 5.2: Best Parameters for All Datasets	108
Table 5.3: Recognition Accuracy (%) For All Datasets	109
Table 5.4: Complexity Parameters for Popular CNNs	109
Table 6.1: Maximum and Average Recognition Rates for ORL Database For All Com- binations of Training Poses	120

Table 6.2: Maximum and Average Recognition Rates for YALE Database For All Combinations of Training Poses	121
Table 6.3: Maximum and Average Recognition Rates for FERET-fc Database For All Combinations of Training Poses	121
Table 7.1: Recognition Time per Folder in seconds for System 2	129
Table 7.2: Recognition Accuracy for System 1	130
Table 7.3: Recognition Accuracy for System 2	131
Table 8.1: Recognition Rate and Comparative Processing Time for BTS, TSRD and GT-SRB Datasets	138
Table 10.1: Recognition Rate of Feature Extraction Methods	157

LIST OF ABBREVIATIONS

1D One-Dimensional

2D DCT Two Dimensional Discrete Cosine Transform

2D DMWT Two Dimensional Discrete Multiwavelet Transform

2D DWT Two Dimensional Discrete Wavelet Transform

2D PCA Two Dimensional Principle Component Analysis

ARA Average Recognition Accuracy

BMP Bitmap

BPNN Back Propagation Neural Network

BSS Blind Source Separation

CRC Collaborative Representation-based Classifier

CSLU Center for Spoken Language Understanding

CVA Common Vector Approach

CWT Continuous Wavelet Transform

DARPA Defense Advanced Research Products Agency

dB Decibels

dB(A) A-weighted Decibels

DCT Discrete Cosine Transform

DFT Discrete Fourier Transform

DL Deep Learning

DNA Deoxyribonucleic Acid

DoD Department of Defense

ELSDSR English Language Speech Database Speaker Recognition

FERET The Facial Recognition Technology

FI Feature Information

FPD Facial Parts Detection

GIF Graphics Interchange Format

ICA Independent Component Analysis

JPEG Joint Photographic Experts Group

KFCG Kekre Fast Codebook Generation

KMCG Kekres Median Codebook Generation Algorithm

KPE Kekres Proportionate Error Algorithm

LBG Linde, Buzo, and Gray

LBP Local Binary Patterns

LBPP Local Binary Probabilistic Pattern

LDA Linear Discriminant Analysis

LL, LH, HL, and HH Low-Low, Low-High, High-Low, and High-High

LPC Linear Predictive Coefficients

LPCC Linear Predictive Cepstral Coefficients

MFCC Mel-Frequency Cepstrum Coefficients

MRA Maximum Recognition Accuracy (for a specific Training-Testing Set)

MRA Maximum Recognition Accuracy

MSE Mean Squared Error

NDSRFR New Discriminative Sparse Representation method for Robust Face Recognition

NIST National Institute of Standards and Technology

NMF Non-negative Matrix Factorization

ORL Olivetti Research Lab

PCA Principle Component Analysis

PDF Probability Density Function

PIN Personal Identification Number(commonly refers to an individual)

PLPCC Perceptual Linear Predictive Cepstral Coefficients

PSO Particle Swarm Optimization

RASTA RelAtive SpecTrAl

RGB Red Green Blue

SSS Small Sample Size

SUSR Short Utterance Speaker Recognition

SVM Supported Vector Machine

TIFF Tag Image File Format

VQ Vector Quantization

DEFINITION OF KEY TERMS

2D-DCT: Two-Dimensional Cosine Transform - expresses a finite sequence of data points as a sum of cosine functions oscillating at different frequencies.

2D-DWT: Two-Dimensional Wavelet Transform - any wavelet transforms for which the wavelets are discretely sampled.

Artificial Intelligence: The theory and development of computer systems able to perform tasks that usually require human logic and intelligence, such as speech recognition, decision-making, visual perception, and translation between languages.

Convolutional neural network: A class of deep neural networks, most commonly applied to analyzing visual imagery.

Dataset: A collection of data.

Eigenvoice: Standardized voice ingredients.

FaceScrub: A Dataset with Over 100,000 Face Images of 530 People.

Facial recognition: A technology able to identify or verify an individual from a digital image.

FERET: (FERET) Dataset used for facial recognition system evaluation as part of the Facial Recognition Technology program.

MATLAB 2018b: Fourth-generation programming language and numerical analysis environment.

Multilayer Perceptron: A class of feedforward artificial neural network.

ORL: Dataset collected by the University of Cambridge that contains a set of face images taken

between April 1992 and April 1994 at the lab.

YALE: Dataset collected by Yale University with 165 grayscale images in GIF format of 15 individuals.

CHAPTER 1: INTRODUCTION

In this work, the focus is on three recognition systems: face, traffic sign, and speaker. All three of these systems belong to the broad field of pattern recognition. Pattern recognition is the classification of data based on features extracted from patterns. Applications of pattern recognition include image processing, computer vision, seismic analysis, radar signal classification/analysis, speaker recognition, and fingerprint identification. One focus of the work in this paper is on the interdisciplinary scientific field of computer vision to enable computers to gain a high-level understanding of the world via digital images; in short, the focus is on automating tasks that currently can only be accomplished by the human visual system. Face recognition is a lively area of research in computer vision; it is a recognition technique used to identify individuals' faces through images stored in a data set. Face recognition systems may be used by law enforcement officers to identify people in photos, videos, or in real-time through their mobile devices to identify people during police stops. Speaker recognition is also studied, which is the automated process of recognizing a person based on their voice and speech patterns. Speaker recognition is the main task in the more general area of speech processing, which has received much focus and seen remarkable progress in the past decades. Companies such as Amazon, Google, Microsoft, and Facebook are investing a large amount of money nowadays in computer vision research and product development, and banks are investing much money in speech processing for recognizing customers over the phone ¹ ².

Many applications depend on the recognition of objects in images to elucidate salient aspects of the world. Meanwhile, language is the engine of civilization, with speech being its most powerful

¹In this chapter; we partially use the material published in the IEEE International Midwest Symposium on Circuits and Systems conference papers, 2019 [2], 2012 [3], 2013 [4], 2018 [5], 2018 [6]

²Papers that are in preparation are [7], [8], [9], [10], [11]

and natural form. The visual and auditory recognition systems considered in this paper hold the potential to make our environment easier to understand, improve the standard of living, and overall enhance our experience of the world.

1.1 Visual Recognition

Digital image processing can be divided into classes such as image enhancement, image restoration, image analysis, and image compression [12]. Image analysis extracts of meaningful information from digital images through digital image processing techniques [13]. Image analysis has been called machine vision, pattern recognition, image description, and computer scene analysis. Examples of image analysis are image processing, feature extraction, and texture analysis. The general steps to image analysis are shown in Fig. 1.1.

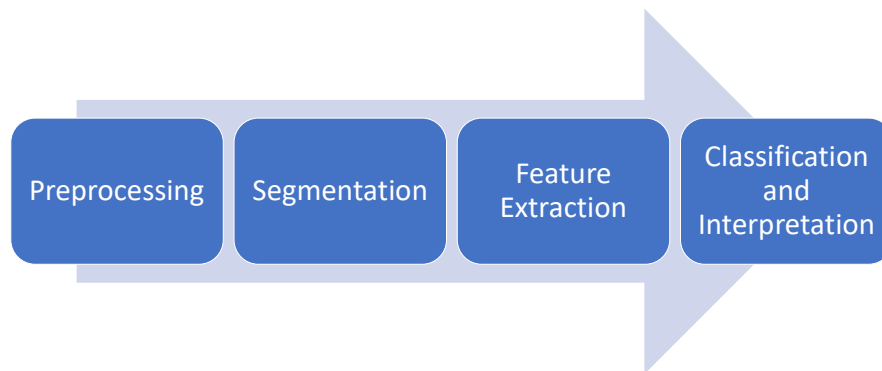


Figure 1.1: Image Analysis Steps

Image analysis techniques can be divided into three categories: feature description, segmentation, and classification. [1] outlines the three categories in Table 1.1. In this dissertation, the two

methods that will be explored for face recognition and traffic sign recognition will be the mix transforms feature extraction methods and neural networks.

Table 1.1: Image Analysis Categories [1]

Feature Description	Segmentation	Classification
Spatial Features	Thresholding	Clustering
Transform Features	Boundary Based Segmentation	Statistical Classification
Edges and Boundaries	Region-based Segmentation	Decision Trees
Shape Features	Template Matching	Neural Networks
Moments	Texture Segmentation	Similarity Measures
Texture		

1.1.1 Applications of Face Recognition

Facial recognition is a Biometric Artificial Intelligence-based technology that can recognize an individual by analyzing patterns found in their facial surfaces and shape [14]. Facial recognition technology has received much attention in recent years because of its potential use in a vast array of applications in both law enforcement and civilian applications (e.g., government, mobile phone, social media, retailers, airlines, and marketers) [15].

1.1.2 Face Recognition Advantages

There are several advantages to face recognition systems, which are considered to be biometrics-based. These systems have several different advantages: the face cannot be forgotten compared with passwords for entry into a building or computer access, the face cannot be easily altered, faces

have distinguishable features, and constructing an algorithm to recognize people based on their faces has a high chance of success. As technology advances, face image acquisition is becoming more accessible, and storage is becoming more abundant.

1.1.3 Issues and Concerns with Face Recognition

The main issues and concerns related to face recognition systems are slow response rate and misidentification of individuals. If the recognition system is being used in real-time, then the time from image acquisition to the recognition needs to be fast. In terms of security, misidentifying an individual can cause trouble by either giving an unauthorized person access to a secure area or prohibiting someone from accessing an area where they should have access. A fast face recognition system can resolve these two issues through speed and high recognition accuracy. Reducing complexity will also speed up recognition. Another issue is storage space; if the system cannot be stored in a mobile device, this limits portability and usefulness of the system. The proposed systems are fast, accurate, low in complexity, and have small storage requirements.

1.1.4 Applications of Traffic Sign Recognition

Rules and regulations are created for highway and roads to keep the general driving public safe. Traffic signs provide information about traffic rules, road conditions, and route directions while assisting drivers for better and safer driving [16]. A car driven outside the traffic rules means a decrease in the safety of the driver, the passengers, and pedestrians. For self-driving cars, it is of utmost importance that the car is driven safely and by the rules of the road. Accurate and fast recognition of traffic signs is the core of the design of autonomous vehicles, and data retrieved from traffic signs, toll information, and vehicle plate numbers are useful in evaluating the surrounding area [17]. The traffic sign recognition systems proposed in this paper are fast and accurate so that

the self-driving car can make a decision based on the recognized image and therefore follow the rules of the road.

1.1.5 Issues and Concerns with Traffic Sign Recognition

There is a vast array of traffic signs which can vary significantly depending on country and region; therefore, datasets containing the various images can be relatively large. At present, several methods have achieved high accuracy. However, this accuracy often comes at the expense of long processing time, which is a debilitating factor in real-world applications of traffic sign recognition and cannot be ignored [2]. When the essential features of the images are extracted and the original dimensions of the images made smaller, the classification time is shortened, which leads to shorter recognition processing time. The shorter recognition processing time means that the sign can be more quickly recognized and deciphered, and therefore the car can more quickly react and consequently follow the rules of the road more closely. By carefully following the rules of the road, the vehicle is driven more safely. Several algorithms have been designed here to more accurately and quickly recognize traffic signs.

1.1.6 Issues and Concerns with Over-Designed Networks for Image Recognition

Image recognition systems are critical components in numerous applications, often requiring real-time implementations that are both fast and accurate. Convolutional neural networks (CNNs) are a standard tool used to meet these conditions. However, with the capabilities and flexibility of a CNN raises the risk of an overly complicated structure resulting in slow processing times and high storage requirements. In image recognition, CNNs are often over-designed to fit general image datasets, thus including more layers and nodes than are warranted by the image features found in a particular application. Two specific applications were studied: facial recognition and traffic sign

identification, and both were analyzed to determine the best characteristics and layout of a CNN that will allow for an adequate fit for the particular datasets.

1.2 Speaker Recognition

Speaker recognition is one of the speech processing fields. Figure 1.2 shows a few areas of speech processing and how speaker recognition relates to the rest of the fields [18]. Speaker recognition systems attempt to recognize a speaker based on features extracted from individual information in a speech signal. It is a biometrics system that is an automated method for verifying a person's identity based on physiological characteristics like voice. Speaker recognition software can be used to control access to restricted services such as phone access to a bank, database services, shopping or voice mail, and access to secure equipment. Speaker recognition can be divided into two components: speaker identification and speaker verification. Speaker identification identifies the speaker that has most likely spoken from a group or population based on the features of a speech signal. Speaker verification accepts or rejects the identity of the speaker based on the sample speech signal. Speaker recognition can be further divided into text-dependent and text-independent systems. A text-dependent system is limited to equivalent spoken speech for both training phases and testing phases, and a text-independent system has no limitation to what is spoken in both training and testing phases.

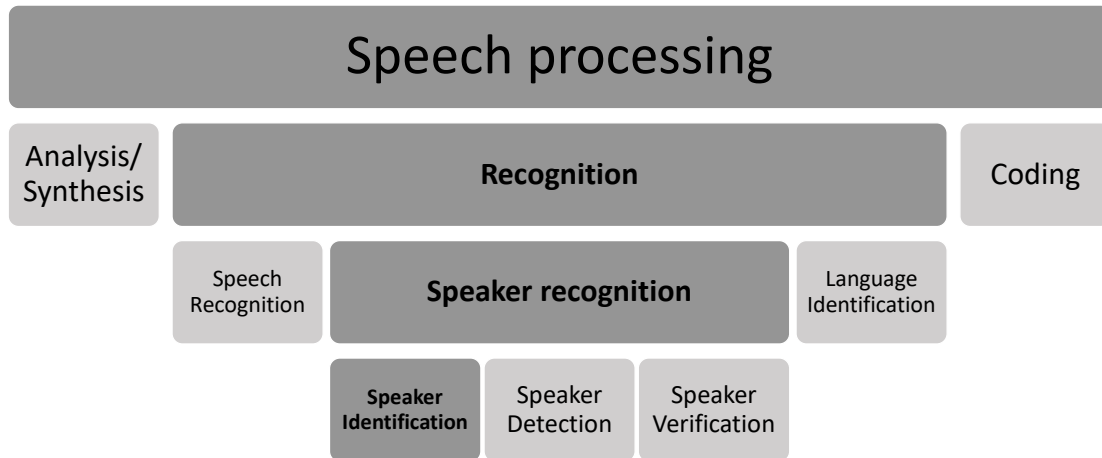


Figure 1.2: Speech Processing

Speaker recognition systems attempt to recognize a speaker based on features extracted from a speech signal. Features like vocal tract system characteristics, pitch, and information patterns in a text convey speaker information [19]. Short-utterance speaker recognition and low memory computations have been a focus of interest in many research investigations. In sentences, vowels have a distinct perceptual advantage over consonants in determining intelligibility and are essential in speaker recognition because they do not require much information from the user.

1.2.1 Applications of Speaker Recognition

Speaker recognition is the process of recognizing a speaking person through speaker-specific information in their speech waves. This information is used to verify identities being claimed by people accessing systems and enables access control of various services by voice [20]. Speaker recognition has been used to ease travel bookings [21], in banking [22], and criminal investigations [23].

1.2.2 Issues and Concerns with Speaker Recognition

Current speaker recognition systems have achieved rather good results; however, critical robustness issues still need to be addressed, especially in practical situations [24].

1.3 Datasets

1.3.1 Face Datasets

Facial recognition belongs to the broader field of biological recognition technology. In the era of digital information, with stricter identification criteria, facial recognition technology is becoming more relevant to humans [25]. The need to better recognize faces is becoming more and more important in fields such as information security, access control, financial payment, and police investigators. These fields can contain thousands of different images that need to be recognized, so there is a need for recognizing faces in large databases. Existing traditional identification approaches for human face detection is very difficult because, in some cases, the human face cannot be captured [26], [27].

Facial recognition has been studied for more than thirty years, and it is one of the most significant areas in the field of pattern recognition and machine vision. Mainly, facial recognition technology under well-controlled imaging condition performs well [20], [21]. However, when the environment cannot be controlled well (i.e., a variation of head poses and changes in illumination), facial recognition accuracy results are poor.

1.3.1.1 Olivetti Research Lab (ORL) Database

The AT&T Laboratories Cambridge database of faces [28] (formerly known as the ORL Database of Faces) contains 40 individuals with ten poses. Each image's pose has the dimensions of 112×92 pixels. The poses vary in position, rotation, scale, and expression as well as in every ten samples, some have open or close eyes and are smiling or not smiling.

ORL database consists of 400 face images captured between April 1992 and April 1994. The database was used in the framework of a facial recognition project carried out in collaboration with the Speech, Vision, and Robotics Group of the Cambridge University Engineering Department(AT&T Laboratories Cambridge). Each person has ten different poses. The lighting conditions were changed against an indistinct homogeneous background with the subjects facing the lens in an upright position. Different facial expressions include closed or open eyes, smiling or not smiling. Some of the images have subjects that are wearing glasses. The image size is 92×112 pixels, with 256 gray levels per pixel. Sample images of the ORL dataset are shown in 1.3.



Figure 1.3: Sample Images from the ORL Dataset

1.3.1.2 YALE

The Yale Face Database [29] (size 6.4MB) contains 165 grayscale images of 15 individuals. There are 11 image poses per person, one for different facial expressions or configuration. For example, center-light, with glasses, happy, left-light, without glasses, normal, right-light, sad, sleepy, surprised, and wink. Sample images of the YALE dataset are shown in Fig. 1.4.

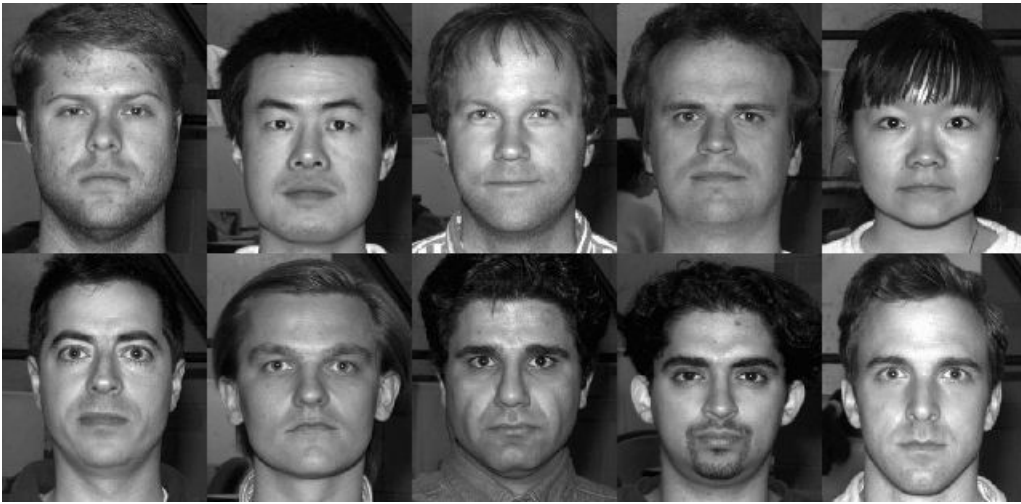


Figure 1.4: Sample Images from the YALE Dataset

1.3.1.3 The Facial Recognition Technology (FERET-fc)

The purpose of the Facial Recognition Technology (FERET-fc) [30][31] program is to develop new techniques and algorithms to recognize human faces automatically. This dataset is sponsored and developed by the DOD Counter-drug Technology Program. The Technical Agent for distribution of the FERET-fc database is the National Institute of Standards and Technology (NIST). As part of the FERET-fc program, this dataset of poses was gathered between December 1993 and August 1996. In this dissertation, the subset fc of the FERET-fc database is used, which consists of 200

subjects with 11 different poses per subject. The resolution of each pose is 256×384 pixels. Hence, the total number of poses in the dataset is 2200. Facial expressions were varied, and face rotations were also considered. Some subjects also wore glasses. Sample images of the FERET-fc dataset are shown in Fig. 1.5.

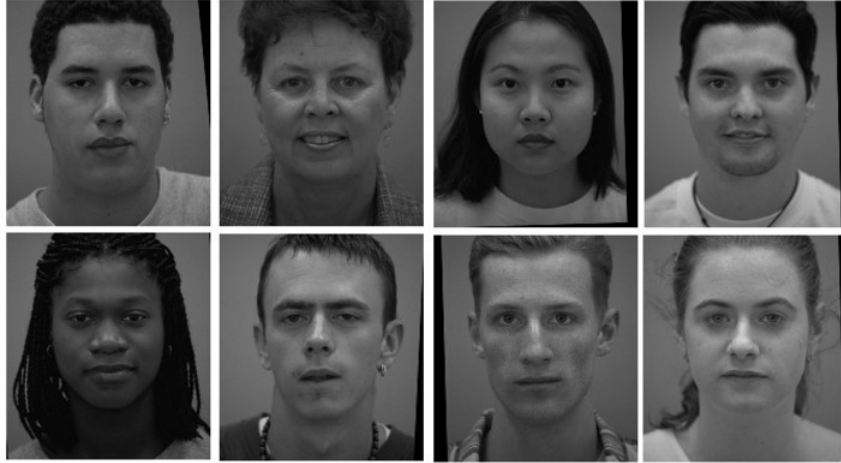


Figure 1.5: Sample Images from the FERET-fc Dataset

1.3.1.4 FEI Dataset

The FEI face database [32] is a Brazilian face dataset containing a set of face images. These images were taken between June 2005 and March 2006 at the Artificial Intelligence Laboratory of FEI in São Bernardo do Campo, São Paulo, Brazil. Every 200 individual has 14 images each for a total image count of 2800. All images are colored and taken against a white homogenous background in an upright frontal position with profile rotation up to 180 degrees. Scale varies around 10%, and the image's original size is 640×480 pixels. All faces are mostly represented by students and staff at FEI, between 19 and 40 years old, with a distinct appearance, hairstyle, and adornments. Male and female subjects are the same and equal to 100. Figure 1 shows some examples of image

variations from the FEI face database. Sample images for the FEI dataset are shown in Fig. 1.6.



Figure 1.6: Sample Images for FEI Dataset

1.3.2 Object Datasets

1.3.2.1 Tiny Imagenet

Tiny ImageNet is a dataset used in the Tiny Imagenet challenge for the default course project in Stanford CS231N class. This challenge runs similar to the ImageNet challenge (ILSVRC), and the goal of the challenge is to get as high as possible accuracy for the image classification problem.

The Tiny Imagenet dataset has 200 classes with each class consisting of 500 training images, 50 validation images, and 50 test images. The site has released the training and validation sets with images and annotations as well as provided class labels and bounding boxes as annotations.

1.3.3 Traffic Signs Datasets

1.3.3.1 BelgiumTS - Belgian Traffic Sign Dataset (BTS)

The paper used cropped images ($\sim 60 \times \sim 80$ pixels) from the BelgiumTSC for classification dataset as a source for street sign images. As Mathias [33] states, their large amount of annotations motivates the choice of the datasets, diversity of the content and classes, the availability of a split for benchmarking traffic sign detection, and traffic sign classification separately. All the data was collected by driving a van with eight cameras on its roof through the streets. About every meter, each of the cameras simultaneously takes a 1628×1236 image. The average speed of the van is $\sim 35\text{km/h}$. Only traffic signs captured at a distance of fewer than 50 meters are considered [33]. The train and the test datasets consist of 62 categories in total. They were combined, and a program was written to gather information about each of the images. Sample images of this dataset are shown in Fig. 1.7.



Figure 1.7: Sample Images from the BTS Dataset

1.3.3.2 GTSRB

The German Traffic Sign Benchmark is a multi-class, single-image classification challenge held at the International Joint Conference on Neural Networks (IJCNN) 2011. There are more than 40 classes of images and more than 50,000 images in total in the dataset. The physical traffic sign instances are unique within the dataset, meaning that each real-world traffic sign only occurs once. The images contain one traffic sign each; the images contain a border of 10% around the actual traffic sign. The images are stored in PPM format (Portable Pixmap, P6). The image sizes vary between 15×15 to 250×250 pixels and are not necessarily square. The traffic sign in the image is not necessarily centered. Sample Images of the GTSRB dataset is shown in Figure 1.8.



Figure 1.8: Sample Images from the GTSRB Dataset

1.3.3.3 *TSRD*

The Chinese traffic sign databases (TSRD) include traffic sign recognition database, traffic sign detection database, and traffic panel database. The camera collects all images in the databases from where the traffic sign stands or from BAIDU Street View. The TSRD includes 6164 traffic sign images containing 58 sign categories. The images are divided into two sub-databases as a training database and a testing database. The training database consists of 4170 images, while the testing one contains 1994 images. All images are annotated using the four coordinates of the sign and the category. Sample Images for the TSRD dataset is shown in Fig. 1.9.



Figure 1.9: Sample Images for TSRD Dataset

1.3.4 Handwritten Digits Dataset

1.3.4.1 MNIST

The MNIST dataset [34] consists of handwritten digits. The training set has 60,000 examples and a test set of 10,000 examples. It is a subset of a more extensive dataset available from NIST. The digits are size-normalized and centered in a fixed-size image. Yann LeCun compiled the dataset from Courant Institute, NYU, Corinna Cortes, from Google Labs, New York, and Christopher J.C. Burges, from Microsoft Research, Redmond. It is a useful dataset for researchers who want to learn techniques and pattern recognition methods on real-world data. Sample Images from the MNIST dataset is shown in Fig. 1.10.



Figure 1.10: Sample Images for MNIST Dataset

1.3.5 Speaker Recognition Datasets

1.3.5.1 Spoken Language Understanding (CLSU) Speaker Recognition Corpus Data Collection (Version 1.1)

The Speaker Recognition corpus (formerly known as Speaker Verification), consists of telephone speech from about 500 participants. All of the data in this corpus were collected over digital telephone lines. The .wav files contain speech data and use the RIFF standard file format. This file format is 16-bit linearly encoded. Each participant repeated the number of strings displayed in Table 1.2 four times during each recording session (for a total of 24 utterances):

Table 1.2: CLSU Speech Data Utterances

Utterance	Used?
5 3 8 2 4	yes
6 1 oh 9 7	yes
4 0 7 1 3	
2 8 3 7 6	yes
1 9 oh 5 4	yes
0 5 2 3 9	

1.3.5.2 *TIMIT*

The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus (TIMIT). Train and Test Data and Speech Header Software - NIST Speech Disc CD1-1.1 - October 1990. Prepared at the National Institute of Standards and Technology (NIST) with sponsorship from the Defense Advanced Research Projects Agency - Information Science and Technology Office (DARPA-ISTO). Text corpus design was a joint effort among the Stanford Research Institute (SRI), Massachusetts Institute of Technology (MIT), and Texas Instruments (TI) TIMIT. The database contains 6300 sentences in total, where each speaker speaks ten sentences. There are 630 speakers from eight major dialect regions of the United States. File types includes .phn (time-aligned phonetic transcription). Table 1.3 shows an example of a sentence expressed phonetically. 100 speakers were tested, nine vowels were used, six samples of the vowels were tested, the best two out of three were matched, single vowel matches tested as well. The sampling frequency is 16 kHz.

Table 1.3: Timit Dataset Speech Phonetic label

She had your dark suit in greasy wash water all year				
0 7470 h#	18950 21053 aa	30960 31870 gcl	43120 43906 w	
7470 9840 sh	21053 22200 r	31870 32550 g	43906 45480 ao	
9840 11362 iy	22200 22740 kcl	32550 33253 r	45480 46040 dx	
11362 12908 hv	22740 23360 k	33253 34660 iy	46040 47480 axr	
12908 14760 ae	23360 25315 s	34660 35890 z	47480 49021 q	
14760 15420 dcl	25315 27643 ux	35890 36971 iy	49021 51348 ao	
15420 16000 jh	27643 28360 tcl	36971 38391 w	51348 52184 l	
16000 17503 axr	28360 29272 q	38391 40690 ao	52184 54147 y	
18540 18950 d	29272 29932 ih	40690 42290 sh	56654 58840 axr	
*eh notshown	29932 30960 n	42290 43120 epi	58840 61680 h#	

1.3.5.3 NOIZEOUS

Noisy speech corpus (NOIZEUS) [35] developed to facilitate the comparison of speech enhancement algorithms among research groups. The noisy database contains 30 IEEE sentences from three male and three female speakers corrupted with different noise at different SNR. These sentences are corrupted by eight different real-world noises such as suburban babble, car, train noise, restaurant, exhibition hall, street, airport, and train-station noise taken from the AURORA database. This corpus is free for researchers. The speech was removed and what was left was the noise for testing. The algorithm was tested using all eight suburban noise as well as white noise.

1.4 Organization of the Dissertation

1. Chapter Two: Presents a literature review of the approaches and algorithms that have been developed for the speaker and image recognition task.
2. Chapter Three: Details and explains several approaches and techniques that were implemented in the research, such as anisotropic diffusion, 2D-DWT, 2D-DCT, neuroevolution (NE), CNN, multilayer perceptron neural networks(MLPNN), and multilayer sigmoid neural networks(MLSNN).
3. Chapter Four: An alternative face recognition system that reduces overall computational complexity by using a few simple algorithms and transforms. The grayscaling algorithm enhances the image, and the salient features are extracted using a mix of two transform families: the two-dimensional discrete wavelet transform (2D-DWT) and the two-dimensional discrete cosine transform (2D-DCT). This combination exploits the nonorthogonality of the coefficients in both domains to preserve the essential details and perceptual qualities of the original image. A multilayer sigmoid neural network is used for classification since the expensive training stage can be performed offline. The trained network, which uses efficient computations, can be embedded in an online system for rapid classification. This system uses grayscaling to enhance the images, as well as a mix of two transform families to extract the salient features: the two-dimensional discrete wavelet transform (2D-DWT) and the two-dimensional discrete cosine transform (2D-DCT). A multilayer sigmoid neural network is used for classification.
4. Chapter Five: A neuroevolution algorithm is developed to reduce the complexity of a CNN architecture by determining the minimal CNN structure and hyperparameters needed to fit a particular dataset adequately. Two specific applications are considered: facial recognition and traffic sign identification. A neuroevolution algorithm is employed to tune the CNN's

parameters and topology to enable a more efficient parameter space search.

5. Chapter Six: A face recognition system that utilizes the DWT and the DCT in sequence and iteratively to best find the features that represent the facial image using the L1 norm classifier for classification.
6. Chapter Seven: A traffic sign recognition system that tests seven of the most popular convolutional neural networks for eight different types of pictures with different tilt, focus, and shade.
7. Chapter Eight: A traffic sign recognition system consisting of normalization, feature extraction, compression, and classification. The images are normalized through gray scaling and anisotropic diffusion. The discrete wavelet and cosine transform extract the essential features of the images as well as decrease the size. A three-layer feedforward multilayer perceptron completes the analysis and classification.
8. Chapter Nine: Presents an original speaker recognition system that utilizes a quantized spectral covariance matrix on the input to a two-dimensional Principal Component Analysis (2DPCA) function. The system is robust in a noisy environment with recognition rates as high as 92% at 0dB Signal to noise ratio.
9. Chapter Ten: Compares and contrasts window frames algorithms. The different feature extraction methods compared are Real Cepstral Coefficients (RCC), Mel Cepstral Coefficients (MFCC), Perceptual Linear Predictive Cepstral Coefficients (PLPCC, and Linear Predictive Cepstral Coefficients (LPCC). The feature extraction methods are used in conjunction with a Vector Quantization (VQ) method and a Euclidean distance classifier to find the best recognition rate among the feature extraction features.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

Over the last few decades, several hundreds of improvements to recognition systems and algorithms have been proposed. Three main areas of improvement are in the preprocessing, feature extraction, and classification modules. The main goal is to have better accuracy while having fewer storage requirements, less computational complexity, and fast processing speed. Nevertheless, the main modules do not have to be complicated, where the lack of complexity leads to diminished storage requirement, complexity, and increase in speed¹.

2.2 Transform Domain

Transform theory is the study of transforms; it is the theory that by using a suitable choice of basis vector space, a problem can be simplified or diagonalized as in spectral theory [36]. That is, transform theory relates a function in one domain to another function in a second domain. One transform that was initially intended for image compression but has seen far-reaching uses in signal processing is the two-dimensional discrete cosine transform (2D-DCT), which is the 2D transform of the DCT. A DCT expresses a finite sequence of data points in terms of a sum of cosine functions oscillating at different frequencies [37]. Much research has been done on DCT, and many variants have been developed for current applications in the medical field, image encryption. [38] developed new image encryption based on singular value decomposition (SVD), fractional discrete cosine transform (FrDCT), and the chaotic system for the security of medical image. [39] proposes

¹In this chapter; we partially use the material published in the IEEE International Midwest Symposium on Circuits and Systems conference papers, 2019 [2], 2012 [3], 2013 [4], 2018 [5], 2018 [6]

a distortion robust DCT-Net, a Discrete Cosine Transform based module integrated into a deep network that is built on top of VGG16. [40], a discrete cosine transform compressed segmented beat modulation method (SBMM) is proposed and its applicability in the case of ambulatory ECG monitoring. [41] proposes an algorithm for the compression of audio signals using DCT with temporal auditory masking (TAM). [42] proposes a ghost imaging method using weight coefficient matching based on DCT, in which the high-quality target images can be retrieved by obtaining the larger weight value in a one-dimensional (1D) DCT spectrum. [43] proposes an approach of compensating the effect of light variations using fractional discrete Cosine transform (FrDCT) to efficiently solve the problem of robust person identification using face images under varying light conditions. All of these proposed DCT methods have a very different configuration, then the proposed systems described in this paper.

The discrete wavelet transform (DWT) has a crucial advantage over Fourier transforms in terms of temporal resolution: that is, it captures both frequency and location information. The two-dimensional discrete wavelet transform (2D-DWT) is the 2D transform of the DWT. DWT is any wavelet transform where the wavelets are discretely sampled. The current research that uses DWT is broad and varied; some of the research is in watermarking, medical applications, fault detection. For example, [44] proposes a digital watermarking method for every type of document image by decomposing the document image into n levels of frequency channels by using Discrete Wavelet Transform (DWT) and embedding the QR code in a sub-band of the final level of DWT. [45] proposes a new method to detect breast cancer using MRI images that are preprocessed using a 2D Median filter, and the features are extracted from the images using discrete wavelet transform (DWT). [46] proposes a High Impedance fault detection method is developed for Smart Distribution Grids based on the Discrete Wavelet Transform (DWT). [47] proposes a new technique for fault classification and detection in the transmission lines of micro-grids using a DWT and a Backpropagation Neural Network (BPNN).

In combination, the DWT and DCT are currently being researched for watermarking, message embedding, and face recognition. In [48], DWT, DCT, bacterial foraging optimization (BFO), and particle swarm optimization have been applied and analyzed for their performance for watermarking medical images. [49] uses a DWT and DCT for the message embedding and extraction stage. [50] proposes an algorithm for face recognition under varying illumination by extracting robust features using DWT with DCT as the feature extraction technique.

2.3 Neural Network Approach

A neural network (currently known as an artificial neural network) is a network or circuit of artificial neurons or nodes [51]. There are several types of artificial neural networks, and each is different based on the implemented mathematical operations and a set of parameters needed to create the output. Feedforward Neural Network (FFNN), Radial basis function Neural Network, Kohonen Self Organizing Neural Network, Recurrent Neural Network(RNN), Convolutional Neural Network (CNN), Modular Neural Network [52] are all neural network examples. The FFNN and CNN were used in our research and will be discussed below.

2.3.1 *Multilayer Sigmoid Neural Network*

In the classification stage, which performs the final step of image classification to recognize individuals, a MLSNN can be used. These MLSNNs are accurate, faster than convolutional neural networks, and less storage-intensive, and have become a standard robust feature extraction method for medical classification [53]. This approach has many of the advantages of neural networks, including the use of simple, homogeneous computations that are easy to parallelize on hardware. It also bears strong performance due to end-to-end training that formulates the problem as a sin-

gle optimization solution, where all components of the model share the same end objective. The idea is to replace the most popular method of using the Euclidean distance for classification as in [54, 55, 56]. [57] has shown that using a simple neural network like the MLSNN provides a much better correlation between the input and the output than the popular Euclidean classification.

The MLSNN is a class of feedforward artificial neural network that is composed of layers of neurons (nodes) where the computations occur. The MLSNN used in this research will have three layers: an input layer, a hidden layer, and an output layer. These nodes are activated when provided sufficient stimulus. This stimulus consists of input from the images, which is multiplied by a set of weights and biases. The nodes amplify the inputs, which will better classify the image, and dampen those that will not. The network is trained using forward and backward propagation known as an epoch.

The MLSNN is a specific type of MLPNN that uses sigmoid functions for the neurons. MLPNNs can have various numbers of layers, and these layers are densely connected. Multilayer sigmoid neural networks have been investigated for various applications. They were first used for pattern recognition [58] and then used to increase the quality of a human face image in [59]. More recently, MLSNNs were used in face gender recognition [60], automatic extraction of the eye and mouth fields from a face image [61], and human face detection in still images [62]. The work proposed here exploits the increasing quality advantage in the pattern recognition of the MLSNN. The work is different in that it recognizes a person from the images, and does not solely categorize female and male subjects or extract faces from a still image. The proposed system is different from the above systems in that there is a separate feature extraction stage from that of the neural network (MLSNN) for facial recognition.

2.3.2 *Convolutional Neural Network (CNN)*

CNNs can also be used in the classification stage of the recognition system. CNN has received significant attention and has had a high impact in terms of efficiency and accuracy in recent years, which is partially due to their outstanding behavior in particularly complex supervised learning tasks [63]. The popular CNNs, despite their accuracy, are typically considered computationally expensive, storage-intensive, and slow compared to feature extraction models in facial recognition[64, 65]. The most popular solution to this slow and bulky CNN is to have as simple as possible architecture for the CNN suitable for classification.

In many cases, the layers within the CNN perform automatic preprocessing that achieves high accuracy compared to other methods and has feature extraction and classification built into its algorithm so the overall system can be made simple. Three different types of datasets will be tested with a more simplified and improved CNN: handwritten digit images, traffic signs images, and face images. These will be compared and contrasted while figuring out the best parameters of CNN to produce accuracy comparable or even better results, without further preprocessing or feature extraction.

The principle issue with utilizing a fully connected feedforward neural network on images is that the number of neurons could be exceptionally high even for shallow architectures, making them impractical for applying on images. The basic idea behind convolutional neural networks (ConvNets) is to devise a solution for reducing the number of parameters allowing a network to be deeper with much fewer parameters [66]. Convolutional neural networks or their variants have obtained state-of-the-art accuracy in computer vision applications, e.g. image classification [67], [68], citewu2017compact, citezhang2015accelerating, facial recognition [69], [64], [70] and traffic sign recognition [71], [72], depending on their deep architectures and a large number of parameters, which leads to considerable consumptions in computation and storage resources. Nonetheless,

the enormous calculation and storage requirements restrict CNNs from being used on increasingly broad applications, for instance, smart cell phones because of their constrained computations and memory capacity. Fortunately, deep models are considerably more redundant in weights, filters, channels, and even layers [73].

Convolutional networks are neural networks that use convolution in place of general matrix multiplication in at least one of their layers [74]. However, most CNN that analyzes images require a multitude of nodes and layers that is complicated in terms of calculations and requires a lot of processing power and time. In this paper, the attempt is to overcome this complicated structure by finding the best parameters that make up a CNN. The main contribution of this paper is to present a creative coding plan that will use the evolutionary algorithm to automatically create a CNN that has the most efficient architecture built on the parts of the CNN configurations. Various parts include hyperparameters, the activation functions, and the number of layers.

Evolutionary algorithms belong to the field known as “neuroevolution,” and for almost three decades, they have had many applications in various fields. Its application for designing CNN for various image recognition application is just starting to show interest and needs to be further investigated. Neuroevolution is a subfield within artificial intelligence (AI) and machine learning (ML) that seeks to develop the means of evolving neural networks through evolutionary algorithms [75]. Neuroevolution is making a resurgence. Prominent researchers in the artificial intelligence field are experimenting with neuroevolution, a series of new successes have bolstered enthusiasm, and new opportunities for impact in deep learning are emerging [75].

Several papers use neuroevolution to find the best convolutional neural network. [76] Performs a neuroevolution method based on a genetic algorithm for finding the optimal deep neural networks architecture in terms of hyperparameters, [77] developed a neuroevolution method able of optimizing and evolving CNNs with respect to the classification error and CNN complexity to re-

duce power consumption further .[78] developed a learning strategy based on neuroevolution to design and train optical neural networks.[79] presented an evolutionary metaheuristic search to optimize deep neural networks and train a convolutional neural network (CNN), and [63] explored the application of neuroevolution to the automatic design of CNN topology.

Backpropagation-based Deep Learning (DL) used in CNN and Neuroevolution have different qualities and shortcomings. DL is good at extracting structure from large amounts of data and producing a compacted internal representation of a high-dimensional input. That is, a deep neural network can learn the characteristic features from a wide variety of pictures from a baseball cap to a bird. However, DL does poorly at solving problems with sparse rewards and at exploring many different strategies for solving a problem simultaneously. Neuroevolution can overcome these limitations. The combination of neuroevolution and DL will allow for the DL to make predictions or recognize objects based on a large number of examples and train a small action-selection component using neuroevolution with the pre-trained deep neural network as a back-end.

2.3.3 Neuroevolution

Neuroevolution (NE) is a form of artificial intelligence that uses evolutionary algorithms (EA) to generate ANN, parameters, topology, and rules. Unique features of EAs are their capability of tackling problems where the structure of a solution is extremely complex, ranging from binary trees [32] to directed graphs [33]; and a straightforward capability of parallel evaluations, as all offspring at a given generation can be evaluated at the same time. For this reason, EAs are applied to complicated tasks such as evolving the layout and the weights of an ANN, with techniques such as NEAT (Neuro-Evolution of Augmenting Topologies) [28], HyperNEAT [29], and Convolutional Neural Fabrics [34]. These algorithms evolve different topologies by optimizing the connections inside the networks.

NE has been around for almost 30 years, and much work has been done with methods such as Evolutionary Programming Network (EPNet), NE of Augmenting Topologies (NEAT), and Evolutionary Acquisition of Neural Topologies (EANT). Efficient Market Hypothesis (EPNet), as described in [80], is a network that emphasizes the evolution of artificial neural network (ANN) behavioral links between parents and their offspring in the NE algorithm. This method was tested on diabetes/heart disease/thyroid dataset. NEAT, as described in [81], which evolves neural network topologies along with weights that employ an ordered method of crossover of different topologies, protects structural innovation using speciation and incrementally grows from a minimal structure. [82] describes the EANT as a topology that evolves the structure and weights of a neural network by compactly encoding the genes of a neural network onto a consecutive genome that enables the evaluation of the network without decoding it. To measure the performance of the algorithm, EANT is tested to see if it can evolve the minimum necessary neural structure for a given learning task. These methods have not been thoroughly tested on CNN and against datasets popularly used today. With the new capabilities of faster computer processing, these methods are taking strides on applying these methods to CNN, where the computing time is very high. NE will significantly reduce the time to train a CNN from current methods.

There has been a renewed interest in NE to be used for CNN. Several methods that use NE to make CNN more efficient has just been recent research however all of these methods have only been tested on the MNIST dataset such as [83] constructs an adaptive system that adapts the activation function within a neural network by the help of Genetic Algorithm (GA) technique. The system is tested on the classification of CT brain images and the MNIST handwritten digits dataset. [63] applies NE to the automatic design of CNN topologies and introduces a common framework for this design.

NE is a well-established research area. However, its application to convolutional neural networks is still marginal. The reason is that CNNs involve very complex topologies, with several convolu-

tional layers consisting of various depths and filter sizes, several dense or recurrent layers with a variable number of neurons, and diverse activation functions. Therefore, only in recent years have improvements in computational resources and research advances enabled the development and validation of tools to specifically optimize this type of deep neural network [63]. Because CNN topologies are expensive, NE has shown to be a promising approach to obtain optimal topologies for a given problem. In recent years, NE systems have been applied in convolutional neural networks (CNN) in the fields of handwriting recognition [63] and human activity recognition [63] as well as in many other fields. In Baldominos' paper, batch size, learning rule, and learning rate are used as parameters to be optimized by the evolutionary approach.

2.4 Unsupervised Learning Algorithms

2.4.1 *Vector Quantization*

Vector Quantization (VQ) is the mechanism of taking a large set of feature vectors and building a smaller set of vectors that represents the centroids of the distribution. There are two phases involved in VQ: training (enrollment) phase, and testing (identifying the speaker from a group) phase. In the training stage, N codebooks are created for all N speakers in a group and stored in a database. It is important to note that these codebooks do not superimpose themselves in the feature space. In the testing phase, a group of vectors are created for the unknown speaker and then compared to the N codebooks in the database. The codebook closest to the unknown speaker is then identified as the speaker. The VQ technique has been chosen because it is effective and to have a high accuracy in the recognition rate [84].

2.4.2 Linear Transformation

In 1991 Turk and Pentland developed the Eigenfaces method based on principal component analysis (PCA) for face recognition [85]. For speech, Kuhn et al. first introduced the concept of Eigenvoices that applies PCA to speaker model parameters, “images”, and applying it to the feature parameters. Eigenvoices is a simple approach to extracting information contained in a collection of speaker traits independent of any for-knowledge of features and uses this information to encode and compare individual speakers. In math terms, the principal components of the distribution of speakers that is the eigenvectors of the covariance matrix of the set of speakers, treating a speaker as a vector in a very high dimensional space. The eigenvectors are ordered, each accounting for a different amount of the variation of the speakers[19]. Eigenvoice is effective for speaker recognition because it can determine the speaker and represent voices, originally in the space of large dimensions, in a low-dimensional linear subspace [86].

2.5 Enhancing the Performance of a Facial Recognition System

First, in face and traffic sign recognition systems, improvements start in the preprocessing step of the recognition system. There are a different number of preprocessing techniques that have been reported in the literature, e.g., grayscaling, Bessel transforms, Anisotropic diffusion, to name a few. In this paper, grayscaling and anisotropic diffusion was used to improve the system before feature extraction.

Secondly, improvements in the feature extraction step enhance the performance of the recognition system. For example, the recognition system performs better if the 2D-DWT and 2D-DCT are employed together. Also, using a neural network in conjunction with the 2D-DWT and 2D-DCT improves feature extraction.

Finally, the classifier can contribute to the recognition system and overall performance and recognition accuracy. CNN, MLSNN, MLPNN are examples of neural networks that have been used in this research to produce excellent accuracy, reduce the complexity of the system, and increase processing speed.

CHAPTER 3: METHODOLOGY

3.1 Preprocessing

Preprocessing an image is used to improve the image data or enhance some image features. Grayscale and anisotropic diffusion are the two methods used in this paper¹.

3.1.1 Grayscale

For many image processing applications, color information does not identify important edges or essential features. Grayscale images capture the luminance of the object and provide a substantial amount of information for distinguishing visual features [87]. The images consist of red, green, blue (RGB) images that can be represented by a matrix of size $M \times N \times P$, where P is the number of channels (in this case, $P = 3$, due to the Red, Green, and Blue components of the image). The grayscale image $x_{gray}[m, n]$, where x_{gray} represents the image as a $m \times n$ numeric array, was obtained by eliminating the hue and saturation information while retaining the luminance. The luminance is calculated from the weighted sum of the RGB component values.

$$x_{gray}[m, n] = 0.2989 * R[m, n] + 0.5870 * G[m, n] + 0.1140 * B[m, n] \quad (3.1)$$

By keeping only the luminance of the image, the number of channels is reduced from three (one for each color) down to one, thus reducing the processing time. The coefficients used to calculate grayscale values are identical to those used to calculate luminance in the document Rec. ITU-

¹In this chapter, we partially use the material published in the IEEE International Midwest Symposium on Circuits and Systems conference papers, 2019 [2], 2012 [3], 2013 [4], 2018 [5], 2018 [6].

R BT.601-7 after the numbers are rounded to 3 decimal places. The Rec.ITU-R BT.601-7 is a document that standardizes parameters for studio encoding of digital television at standard 4:3 and wide-screen 16:9 aspect ratios.

3.1.2 Image Resize

Image resize used the nearest-neighbor interpolation. The nearest neighbor method is a statistical test that is used to determine the significance of a point's nearest neighbor in order to calculate the deviation from the general trend. The nearest neighbor algorithm selects the nearest point value and does not take into account the values of other neighboring points, hence producing a constant interpolation [88]. The images were resized to 64×64 to give enough information for each image and to be able to represent an image best to distinguish it from a dataset that has many categories (poses).

3.1.3 Anisotropic Diffusion

Illumination normalization can be achieved through a simplified Perona-Malik Anisotropic Diffusion [89]. The image $I_0(x, y)$ is the input image normalized iteratively until the edges are accurately detected. The normalized image, $I(x, y)$, is expressed in Equations 3.2 and 3.3.

$$I_{n+1}(x, y) = \frac{\sum_{i=-1}^1 \sum_{j=-1}^1 w_{ij} I_n(x + i, y + j)}{\sum_{i=-1}^1 \sum_{j=-1}^1 w_{ij}} \quad (3.2)$$

$$w_{ij} = e^{-k|I_n(x,y)-I_n(x+i,y+j)|} \quad (3.3)$$

Through [90] criteria, the number of iterations and k was found to be 2 and 0.4, respectively, in this dissertation.

The images were also normalized using anisotropic diffusion to reduce image noise without removing significant parts of the image content such as edges, lines, and other details that are important for the interpretation of the image [91].

3.2 Image Feature Extraction

Feature extraction is crucial to the identification of the person as the input to the image model and pattern matching process. Two favorite transforms used for extracting features are the DWT and DCT. The DWT and DCT are both Fourier-related transform. The DWT expresses an image in terms of a sum of wavelets (a mini-wave). The DCT expresses an image in terms of a sum of sinusoids with different frequencies and amplitudes. Preprocessing an image is often used to improve the image data or enhance some image features. The key idea in this dissertation is to pass extract features using the 2D-DWT and the 2D-DCT, which will only keep the dominant coefficients of the image while simultaneously reducing the size of the image. By reducing the number of coefficients processed by the transforms, the computational processing of the images will be faster.

3.2.1 Two-Dimensional Discrete Wavelet Transform (2D-DWT)

The 2D-DWTs have found wide application in signal and image processing [92], for example, in image compression, noise-canceling, and feature extraction. Wavelet transforms novel mathematical techniques based on group theory and square-integrable representations, which allow one to unfold a signal (image) both in space, scale, and sometimes direction [93]. The advantage of using a DWT is that it can represent different regions of an image with different degrees of resolution. Wavelet transforms are widely used for both denoising [94] and compression of images [95]. Wavelet analysis decomposes a signal into a set of basis functions called wavelets. The wavelet's construction is the product of shifting (translation) and scaling (dilation) of the mother function. If a discrete signal has known basis functions and abides by $\{f[n] | \sum_{n=-\infty}^{\infty} |f[n]|^2 < \infty\}$, then it can be approximated as in equation (3.4) from Chun-Lin [96],

$$f[n] = \frac{1}{\sqrt{M}} \sum_k W_\phi[j_0, k] \phi_{j_0, k}[n] + \frac{1}{\sqrt{M}} \sum_{j=j_0}^{\infty} \sum_k W_\psi[j, k] \psi_{j, k}[n], \quad (3.4)$$

where the discrete signal $f[n]$, scaling function $\phi_{j_0, k}[n]$, and mother wavelet $\psi_{j, k}[n]$ are discrete functions defined in $[0, M - 1]$ given a total of M samples to be transformed. This number is typically selected to be $M = 2^K$, where K indicates the number of transform levels. Because the scaling function and the mother wavelet are orthogonal to each other, the inner product can be used to obtain the approximate coefficients $W_\phi[j_0, k]$, and the detailed coefficients $W_\psi[j, k]$ [96].

$$W_\phi[j_0, k] = \frac{1}{\sqrt{M}} \sum_n f[n] \phi_{j_0, k}[n] \quad (3.5)$$

$$W_\psi[j, k] = \frac{1}{\sqrt{M}} \sum_n f[n] \psi_{j, k}[n] \quad j \geq j_0 \quad (3.6)$$

The basis functions $\phi_{j_0,k}[n]$ and $\psi_{j,k}[n]$ are defined as

$$\phi_{j_0,k}[n] = \frac{1}{\sqrt{2^j}} \phi\left[\frac{n}{2^j} - k\right] \quad (3.7)$$

$$\psi_{j,k}[n] = \frac{1}{\sqrt{2^j}} \psi\left[\frac{n}{2^j} - k\right] \quad (3.8)$$

where $\phi[n]$ and $\psi[n]$ are the scaling and wavelet functions.

The DWT is any wavelet transform for which the wavelets are discretely sampled. This transform decomposes the signal into two components using a set of quadrature mirror decomposition filters, with one containing the low-frequency coefficients (referred to as the approximation), and the other containing the high-frequency coefficients (called the details). These components are calculated using equations (3.5) and (3.6), respectively. The input to the transform is a discrete signal $f[n]$ of length M where $M = 2^K$ for some integer K . At the first level, the approximation coefficients, $\phi_1[n]$ are calculated as the convolution of the input signal $f[n]$ with the lowpass filter $h_\phi[n]$. In the same manner, the detailed coefficients $\psi_1[n]$ are calculated as the convolution of the input signal $f[n]$ with the highpass filter $h_\psi[n]$, as shown in equations (3.9) and (3.10).

$$W_\phi[j, k] = h_\phi[-n] * W_\phi[j + 1, n]|_{n=2k, k \geq 0} \quad (3.9)$$

$$W_\psi[j, k] = h_\psi[-n] * W_\psi[j + 1, n]|_{n=2k, k \geq 0} \quad (3.10)$$

The 2D-DWT is a natural extension from the one-dimensional signal domain. 2D-DWT applies the 1D-DWT on each row variable and then on each column of the image matrix. At each step, two sub-images are created, with half the number of pixels found in the processed row or column.

In the end, an image of size $M \times N$ is decomposed into four sub-images of size $M/2 \times N/2$. More precisely, the 2D-DWT decomposition yields an approximation subband (LL), a horizontal detail subband (LH), a vertical detail subband (HL), and a diagonal detail subband (HH). A schematic diagram of a general 2D-DWT is shown in Fig. 3.1.

The approximation band, which contains essential information about the image, consists of low pass filters, $h_\phi(n)$, and downsampling by two. The detail bands are built from a combination of the outputs from the high pass, $h_\psi(n)$, and low pass filters, and consist of details about the diagonal, horizontal, and vertical information. In the proposed feature extraction stage, the approximation band (LL band) is kept, and the detailed bands are discarded. Figure 3.1 shows the image decomposed into the detailed and approximated coefficients.

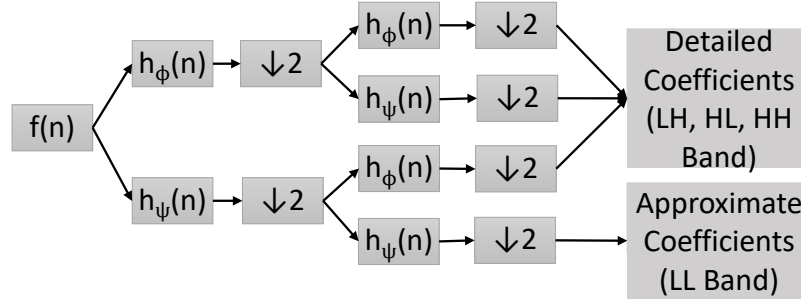


Figure 3.1: Block Diagram of Discrete Wavelet Filter Analysis

There are dozens of wavelet families, and their members are generally defined numerically through the associated filters. The abundance of variants stems from the fact that, even though the fast wavelet transform has a recursive time/space frequency resolution, there are various ways to optimize the time/space and frequency localization. However, not all wavelets perform well in all applications, and for the optimal wavelet to be chosen, the properties of the wavelets must be understood in the context of the given problem. The essential features of the wavelet are the sup-

port size, symmetry, number of vanishing moments, regularity, and (bi-)orthogonality [97]. The properties of the scaling functions are determined by those of the wavelets but are themselves less relevant.

3.2.2 Two-Dimensional Discrete Cosine Transform (2D-DCT)

A deficiency in using a traditional wavelet, such as Haar, in DWT is the complicated calculations. The calculations can be reduced by a quarter or more by reducing the dimensions of the image by applying the 2D-DCT. One of the main advantages of the 2D-DCT (and DCT in general) is its redundancy removal property between neighboring pixels, called a decorrelation characteristic. Due to the smoothness often found in surfaces, neighboring pixels are usually correlated, leading to higher energy compaction in a few low-frequency coefficients of the 2D-DCT frequency pattern [98]. This leads to 2D-DCT excelling in terms of compression [99] and face recognition [100], even under varying face illumination conditions [101].

The 2D-DCT is the two-dimensional extension of the one-dimensional DCT (1D-DCT). That is, the 1D-DCT is performed along the rows and then along the columns. The original image pixel data is labeled $h(x, y)$ with dimensions $M \times N$, and the output DCT coefficient are represented by $H(u, v)$. The equations for calculating the 2D-DCT are shown in (3.11), (3.12), and (3.13).

$$H(u, v) = \frac{2}{\sqrt{MN}} C(u) C(v) \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} h(x, y) \cos(A) \cos(B) \quad (3.11)$$

$$A = \left[\frac{(2x+1)u\pi}{2M} \right], B = \left[\frac{(2y+1)v\pi}{2N} \right] \quad (3.12)$$

$$C(\gamma) = \begin{cases} \frac{1}{\sqrt{2}} & \gamma = 0 \\ 1 & \gamma > 0 \end{cases} \quad (3.13)$$

3.2.2.1 Biorthogonal Wavelets

There are dozens of wavelet families, and their members are generally defined numerically through the associated filters. The biorthogonal wavelet (BOW) will be used in this research. There are many advantages to using the BOW for DWT. First, the BOW has a linear phase property, which leads to the filter coefficients being symmetric. The linear phase is very critical for image processing because the BOW filter will not introduce visual distortions in the image. Second, the BOW is commonly used in image processing to detect and filter white Gaussian noise because of their high contrast of neighboring pixel intensity values [102]. That is, wavelets consist of scaled and translated versions of the mother wavelet, called basis wavelets. Basis wavelets form an orthogonal basis to the mother wavelets. The BOW can be defined solely based on the scaling filter [103]. By using only the basis of the scaling filter, BOWs are capable of compressing the signal such that unwanted data (noise) are eliminated. Consequently, the wavelet preserves the energy of the original image. Third, the BOW is useful at reducing unnecessary features. One primary deficiency that exists in the utilization of a traditional wavelet (i.e., Haar wavelet) is that in the DWT, the Haar wavelet yields a large number of features unnecessary for accurate recognition. The issue of a large number of redundant features can also be resolved by the property of bi-orthogonality in the BOW that is intimately related to the non-redundancy of the BOW transform [97].

3.2.3 Neuroevolution

NE is a machine learning technique that applies evolutionary algorithms (EA) to generate artificial neural networks (ANN), parameters, topology, and rules by taking inspiration from the evolution of biological nervous systems in nature [1]. A representation of a NE process is shown in Fig. 3.2. In an EA, a single candidate solution is called an *individual*; the set of all individuals is called the *population*. Evolution proceeds through discrete steps in a cycle called *generations*. After each generation, the weakest in the population is replaced by a stronger offspring through the processes of breeding known as speciation. Breeding occurs between the top two fittest individuals. This is determined by the fitness of each of the individuals in a population. The fitness function measures the ability of an individual to solve the target problem. In this case, the target problem is to find the best parameters that make up the CNN that gives the highest accuracy. The fitness influences the probability of a solution to propagate its characteristics to the next generation.

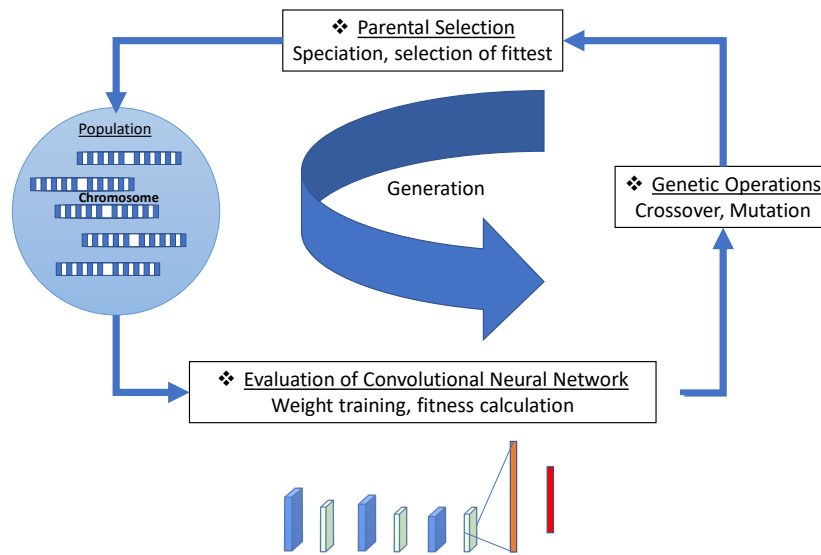


Figure 3.2: Illustration of the Typical Generational Neuroevolution Process

To breed, individuals must have genetic information associated with their fitness. This genetic information is held in what is called a *chromosome*. This chromosome holds the genetic information of the individual. In this case, this genetic information is the characteristics/parameters of the CNN. Each of the characteristics is called a *gene*. There are two main breeding methods: crossover and mutation. During a crossover, a random position in the chromosome is picked (the crossover point), and the gene information from one individual's chromosome is exchanged with the second fittest individual's gene information at the crossover point. This exchange creates a new individual, the *offspring*. Mutation means that the offspring is a mutated version of both parents where one gene is chosen at random, and a new random characteristic is devised for that gene.

The primary limitation of EAs lies in their dependence upon an evaluation function, the function that determines the fitness of an individual. This function is called a considerable number of times during each generation. In this case, the evaluation function is the recognition accuracy determined through CNN with the chosen characteristics/parameters. Because the fitness function is computationally expensive to run, this makes the training of the system computationally expensive. However, once the parameters for the CNN have been determined in training, the testing phase only requires the test image to be processed by the CNN, and the best match is recognized. The benefit of EA is that it is finding the best possible solution with the highest accuracy. It has been shown that EAs can deliver better results than the state-of-the-art in traditional optimization methods based on gradient descent [104].

3.2.4 Convolutional Neural Networks

Convolutional neural networks are very accurate when used for image recognition. However, CNNs, despite their accuracy, are typically considered computationally expensive, storage-intensive, and slow compared to feature extraction models in facial recognition[64, 65]. CNN is considered

to be complex, and the actual design of the network can be determined by trial and error and is very ad hoc in determining the required parameters. Using NE for adjusting the parameters, architecture, or hyperparameters of the CNN is efficient and leads to a much more compact system tailored to the dataset being analyzed.

The overall architecture of CNN's consists of a feature extractor and a classifier. In the feature extraction layers, each layer of the network receives the output from its previous layer as its input. Its output is passed sequentially as the input to the next layer. The CNN architecture consists of a combination of three types of layers: convolution, max-pooling, and classification. The convolution layer learns the basis vectors of images and extracts useful higher-level features through a hierarchical process. Max-pooling is a sub-sampling layer, and the classification is a fully connected traditional multiple layer perceptron. See Fig. 3.3 for the typical framework of a CNN.

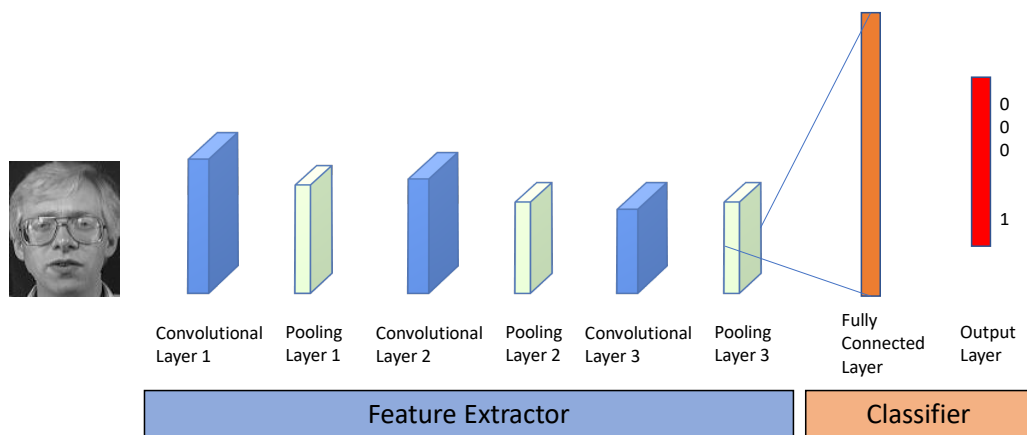


Figure 3.3: Basic Convolutional Neural Network

Feature maps are obtained by convolution from the learned basis vectors of input images [105]. The equation that relates the input image I and kernel to create a feature map x (a parameterized

representation of a surface in the space) where the kernel is a random filter.

$$x_{i,j} = (I * K)_{ij} = \sum_{m=0}^{k1-1} \sum_{n=0}^{k2-1} K_{m,n} \cdot I_{i+m,j+n} + b_i \quad (3.14)$$

$$K_{i,j} = \sum_{m=0}^{k1-1} \sum_{n=0}^{k2-1} RandomNumber[-0.25, 0.25] \quad (3.15)$$

$$b_i = \sum_{m=0}^{k1-1} RandomNumber[-0.25, 0.25] \quad (3.16)$$

Each plane of a layer is usually a derivative from the mixture of one or more planes of previous layers. The nodes of a plane are connected to a small region of each combined plane of the previous layer. Each node of the convolution layer extracts the features from the input images by convolution operations on the input nodes. The basic pseudocode of CNN is shown in 1.

3.2.4.1 Pooling layer

Pooling layers often follow a convolutional layer. The pooling layer takes a small subset from the convolutional layer and produces a single output from the small subset. Pooling is done in several ways: finding the average, the maximum, or calculating a learned linear combination of the neurons in the block. The focus will be on the more popular max-pooling layers; that is, the maximum of a subset of the matrix (kernel) is pooled. The max-pooling layers do no learning; it takes merely a $k \times k$ region and outputs a single value, which is the maximum of that region. For example, if the input layer is a $N \times N$ layer, the output will then be $\frac{N}{k} \times \frac{N}{k}$ layer, as each $k \times k$ block is reduced to a single value by a max function. The pooling layer reduces the height and

width of the input. It helps minimize computation, as well as helps make feature detectors more invariant to its position in the input.

3.2.4.2 Fully Connected layer

After several convolutional and max-pooling layers, the number of feature maps increases to better represent features of the input images and ensuring classification accuracy. The outputs of the last layer of CNN are the inputs to a fully connected network, which is called the classification layer. Fully connected layers are the high-level reasoning in the neural network that ensures the classification. A fully connected layer combines all neurons in the previous layer, which is the fully connected layer, the pooling layer, or the convolutional layer and connects it to every single neuron it has.

It is important to note that features propagate from lower-level layers to derive higher-level features. As the features propagate to the highest level, the dimensions of features are reduced. This reduction depends on the size of the kernel for the convolutional and max-pooling operations, respectively. Note that in the context of convolutional neural networks, a node is a filter which is also known as a feature detector.

3.2.4.3 Forward and Backward Propagation

The architecture of the CNN was a multilayer neural network with four hidden layers (two convolutions/pooling layers and two fully connected layers). The layers are comprised of nodes, which are places where computation occurs, loosely patterned on a neuron in the human brain, which is activated when it meets sufficient stimuli that combine input from the input images with a set of weights that either amplify or dampen the input data — the nodes assigned significance to in-

puts that will better classify the image. The network consists of a forward and backpropagation algorithm to train the network. The input layer serves as the receiving end of the preprocessing (grayscale) and has nodes equal in dimension to the feature vectors. The output layer has one node.

The number of epochs, which is a complete pass through a given dataset, is determined using a cross-entropy loss of 0.001. The training samples are represented by $p_1, n_1, p_2, n_2, \dots, p_Q, n_Q \in \mathbb{R}$, where $p_q = [p_{q1}, p_{q2}, \dots, p_{qR}]$ are inputs to the network. The input to the CNN is a three-dimensional matrix that represents each of the training image samples for each category. If the size of the images is $m \times m$ and there are p categories holding q samples each, then the input matrix will be $m \times m \times pq$.

The forward propagation uses weight initialization, calculation of the activation unit, weight adjustment, weight adaptation, and convergence testing. Initially, all weights are set to small random values ranging between -0.25 and 0.25 . Assume w_{ji} represents the weight between the j th layer and i th layer, and w_{j0} is the initial weight. The activation units are calculated sequentially, starting from the first layer. The activation function for each layer is calculated as follows in equation 3.5:

$$x_j^p = S_{y_j}^p \left(\sum_{i=1}^I \omega_{ji} x_i - w_{j0} \right) \quad (3.17)$$

where a_{pj} is the activation of the j^{th} hidden unit for the pattern. x_i is the input to the layer, and S is the activation function where the rectified linear unit (ReLU), sigmoid, and hyperbolic tangent are the most popular. In our proposed four-layer system, the ReLU and the hyperbolic function were used respectively in the first two hidden layers. The hyperbolic function and sigmoid function were utilized for the two fully connected layers, respectively.

Average Loss Array: Cross-entropy

$$Loss\ Array = -\frac{1}{c} \sum_{c=1}^c [yy_c \cdot \ln(o_c^L) + (1 - yy_c) \ln(1 - o_c^L)] \quad (3.18)$$

Where n is the number of categories, yy is the labels, and oL is the output of the last layer.

Initially, all weights are set to small random values. Assume v_{ji} represents the weight between the j_{th} hidden unit and the i_{th} input unit and w_{kj} represent the weight between the k_{th} output and the j_{th} hidden unit. The activation unit is calculated sequentially, starting from the input layer. The activation of hidden and output units is calculated as follows:

$$y_{ij}^{(p)} = f_{yj}^{(p)}(I * K)_{ij} = f_{yj}^{(p)} \left(\sum_{m=0}^{k1-1} \sum_{n=0}^{k2-1} v_{m,n} \bullet I_{i+m, j+n} + b_{j0} \right) \quad (3.19)$$

$$o_k^{(p)} = f_{ok}^{(p)} \left(\sum_{j=1}^J w_{kj} y_j - w_{ko} \right) \quad (3.20)$$

Where $y_{ij}^{(p)}$ is the activation of the j_{th} hidden unit and $o_{qk}^{(p)}$ is the activation of the k_{th} output unit for the pattern I . f is a *tanh* function.

$$E^P = \frac{1}{2} \sum_{k=1}^K (t_k^{(p)} - o_k^{(p)})^2 \quad (3.21)$$

Where k is the total number of output units, I is the total number of inputs, and J is the total number of hidden inputs.

3.2.4.4 Activation Functions

Neural network activation functions are a critical segment of DL. Activation functions decide the output of a DL model, its accuracy, and its computational efficiency of training a model. Activation functions have a major effect on the neural network's capacity to converge and the convergence speed. In some instances, activation functions might prevent neural networks from converging at all. The relationship between the input, the weights, the activation functions, and output is shown in Fig. 3.4. Equation (3.22) shows the equation relating the above mentioned parameters and (3.23) is the compact vectorized version of the neural network.

$$x_j^l = \sigma \left(\sum_k w_{j,k}^l \cdot x_k^{l-1} + b_j^l \right) \quad (3.22)$$

$$x = \sigma(w^l x^{l-1} + b^l) \quad (3.23)$$

l represents the layer when $l - 1$ is the previous layer to l .

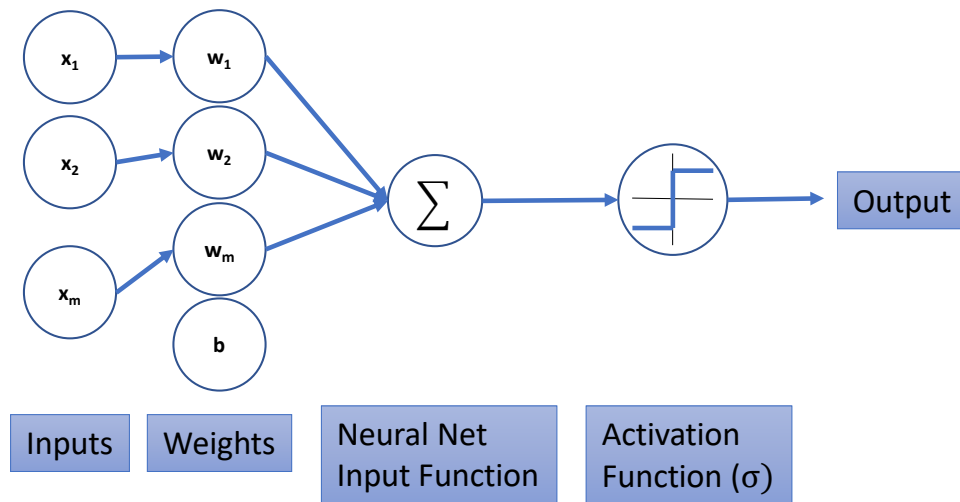


Figure 3.4: Model of Neural Network

There are advantages and disadvantages to the sigmoid activation function. The advantage of this function is that it has a smooth gradient, which prevents “jumps” in output values. The output values are bound between 0 and 1, normalizing the output of each neuron. For x values above 2 or below -2 the prediction $f(x)$, the output of the sigmoid activation function shown in Eq. (3.24), is at the edge of the curve that is very close to 1 or 0 and therefore enables clear predictions. The sigmoid activation function does have disadvantages such as vanishing gradient for very high or very low values of X . This is because there is almost no change to the prediction, causing a vanishing gradient problem. This lack of change can result in the network to fall into a local minimum and fail to learn further, or being too slow to converge to an accurate prediction. Also, the outputs are not zero centered, and this function is computationally expensive. The equation for

the sigmoid activation function and the derivative is shown in Equation (3.24) and (3.25).

$$f(x) = \frac{1}{1 + e^{-x}} \quad (3.24)$$

$$f'(x) = \frac{\partial}{\partial x} f(x) = f(x)(1 - f(x)) \quad (3.25)$$

Concerning the loss function, the cross-entropy is nearly always the better choice, if the output fully connected layer is made up of sigmoid neurons. The reason is that when the network has first initialized, the weights and biases are randomized. These initial choices result in the network being decisively wrong for some training input, which means that the output neuron will have saturated near 1 when it should be 0, or conversely saturated near 0 when it should be 1. If the quadratic cost function is used, then there will be a decrease in learning. The learning will not stop entirely because the weights will continue to learn from other training inputs, but it is at those neurons; it is undesirable.

The Hyperbolic tangent activation function is similar to the sigmoid function but zero centered, making it easier to model inputs that have strongly negative, neutral, and strongly positive values. This function has the same disadvantages as the sigmoid function. The function equation and its derivative are shown in (3.26) and (3.27).

$$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (3.26)$$

$$f'(x) = \frac{\partial}{\partial x} f(x) = 1 - f(x)f(x) \quad (3.27)$$

The ReLU activation function is computationally efficient, which allows the network to converge very quickly. Although the function looks linear, it is non-linear and therefore has a derivative function that allows for backpropagation. There is a disadvantage of the dying ReLU problem when the inputs approach zero or the inputs are negative; the gradient of the function becomes zero. In this case, the network cannot perform backpropagation and cannot learn. To prevent the dying ReLU problem, the leaky ReLU has a small positive slope in the negative area, so it does enable backpropagation, even for negative input values. The disadvantage is that the results not consistent; that is, leaky ReLU does not provide consistent predictions for negative input values. The function equation and its derivative are shown in (3.28) and (3.29).

$$f(x) = 1(x < 0)(\alpha x) + 1(x \geq 0)(x) \quad (3.28)$$

$$f'(x) = (\alpha + (1 - \alpha)(x > 0)); \quad (3.29)$$

where alpha is a small constant representing the leak with a value of 0.01 in our algorithm. Fig. 3.5 and 3.6 compare the activation functions and their derivatives.

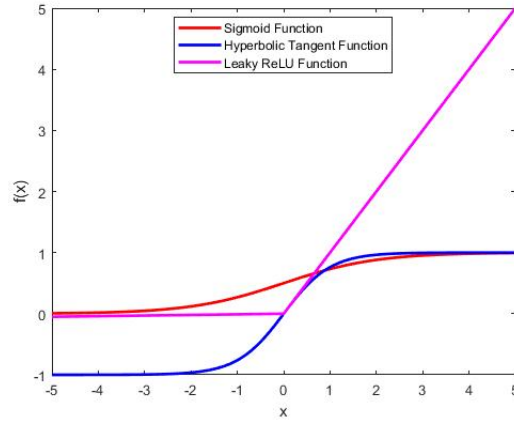


Figure 3.5: Comparison of Activation Functions

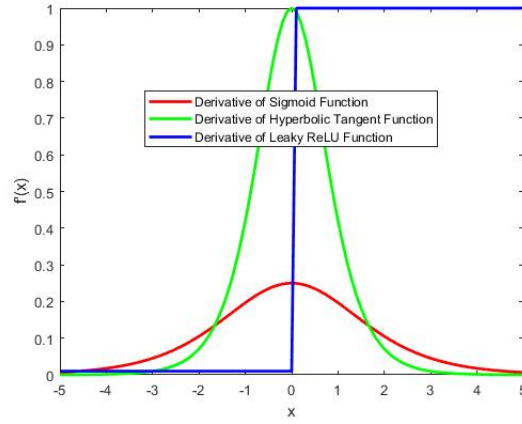


Figure 3.6: Comparison of Derivative of Activation Functions

The loss function, the sum of squares of the absolute errors between the actual output and the desired output, was determined at the output. The backward propagation used differentiation to optimize the weights, which recursively propagated back from the output layer through the hidden layers to the input layer. The minimization of the error E requires the partial derivative of this

error with respect to each weight in the network to be computed. The change in the weights is proportional to the corresponding derivative, where η is the learning rate, which was set to 0.01 in our system. The last term is a function of the previous weight change. At each step, n , the weights are updated. The equation is shown in 3.6.

$$\Delta w_{ji}[n+1] = -\eta \frac{\partial E}{\partial \omega_{ji}} + \alpha \Delta w_{ji}[n] \quad (3.30)$$

Note that w_{ji} and x_{kj} are weight adjustments. Through the process of gradient descent, the process repeats until the desired output is reached. The output of this network is a vector with a weight for each of the categories and testing samples. Neuron bias is added to the layers of the CNN to offset the origin of the activation functions. This bias makes room for rapid convergence in the training process. The bias was set to a random value between -0.25 and 0.25 . In the same manner, for the other weights, the bias weights were trained. The weights were updated in the hidden layers shown in equation 3.7.

$$w_{ji}[n+1] = w_{ji}[n] + \Delta w_{ji}[n+1] \quad (3.31)$$

A CNN leverages the fact that an image is composed of smaller details, or features, and creates a mechanism for analyzing each feature in isolation, which informs a decision about the image as a whole [106]. The key parameters that will be analyzed when designing the proposed CNN are:

1. **Number of Samples:** This parameter depends on the number of objects or poses in a certain category. Normally face datasets have 11 to 15 poses for individuals, traffic sign datasets will have 50 to 150 different images of a certain traffic sign, more general datasets can have up to 1000 different images that represent a certain category.

Algorithm 1: Pseudocode of Typical CNN System

1: Open dataset:

- Open training and testing images
- Open training and testing labels

2: Initialize parameters such as:

- image height and weight
- number and structure of layers
- type of loss function
- learning rate

3: Create CNN:

- Create convolutional layers, pool layers, and fully connected layers
- Initialize the number of epochs (an epoch is when an entire dataset is passed through the neural network forward and backward once)
- batch size (total number of training examples present in a single batch).

4: Train the CNN:

- Iterate through the number of epochs apply the layers of the network (convolution, pooling layers).
- Involves the gradient descent and adjusts the weights.

5: Test images by using the above configured CNN

2. **Batch Size:** The batch size is a hyperparameter that is part of the gradient descent algorithm that controls the number of training samples to analyze before the CNNs internal parameters are updated.

3. **Convolutional Layers/Pooling Layers:** The number of layers represents how many groups of convolutional and pooling layers are required to build the simplest and most effective CNN.

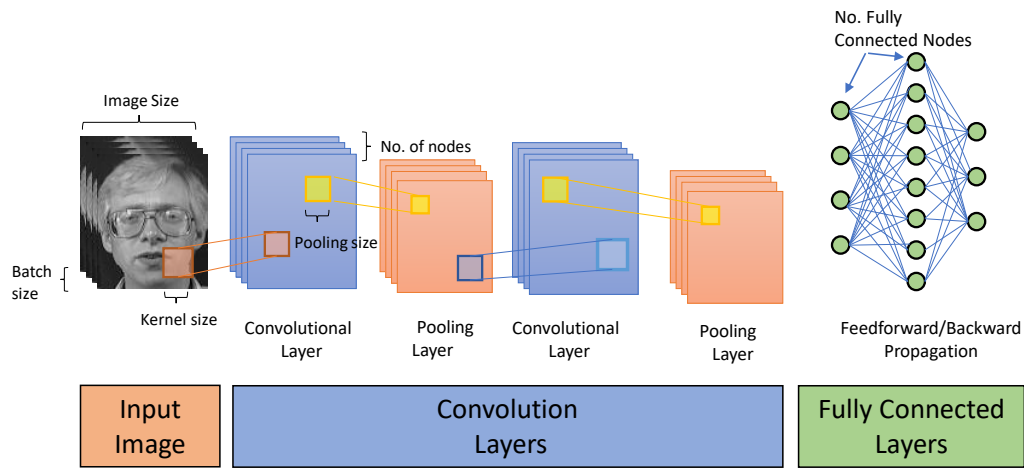


Figure 3.7: Model of Convolutional Neural Network Showing Parameters

4. **Kernel Size:** CNN is a stack of layers that are defined by the action of several filters on the input. These filters are called kernels. The kernel size is the *width* \times *height* of the filter mask. There is three kernel size that will be considered for each of the convolutional layers ($k1$, $k2$, and $k3$).
5. **Number of Nodes:** The kernel is swept across the image as it filters a section of the image. There are slightly fewer nodes than there are input nodes due to zero-padding. The stride which moves the kernel across the image is one pixel at a time in our research. Three nodes will be considered for each layer ($n1$, $n2$, and $n3$).
6. **Cost:** The cost is the point in the iteration (epoch) where the iteration stops to give you the best accuracy.
7. **Number of Fully Connected Layers:** The fully connected layers take the result of the convolution/pooling process and make a classification decision. One or two layers will be the focus of this research. The fully connected input layer takes the output of the convo-

lution/pooling layers and turns the matrix into a single vector (flattens the matrix) that is inputted to the next fully connected layer. The fully connected output layer gives the final probabilities for each label. That classifies the image into its appropriate category.

8. **Number of Nodes in the Fully connected layers:** The nodes in fully connected networks are commonly referred to as neurons. In a neural network, each neuron computes an output value by applying a specific function to the input values arriving from the previous layer. Two possible number of nodes will be considered in this research for each possible fully-connected layer.
9. **Activation Functions:** The function that is applied to the input values determined by a vector of weights and a bias. In a neural network, learning progresses by making iterative adjustments to these biases and weights. Three activation functions will be considered in this research: the rectified linear unit (ReLU), the hyperbolic tangent (tanh), and the sigmoid (sigm) function.
10. **Learning Rate:** The learning rate controls how quickly or slowly, a CNN model learns a problem. CNN uses backpropagation with gradient descent to determine the best weights (for the nodes/neurons), and one of the key hyperparameters to set in order to train a neural network is the learning rate for gradient descent. The learning rate scales the magnitude of our weight updates in order to minimize the network's loss function.
11. **Image Dimensions:** The image dimension is the height and length of the image before it enters the CNN.

3.3 Image Classification

3.3.1 Multilayer Perceptron Neural Network (MLPNN)

MLPNN is a relatively simple neural network that achieves state-of-the-art accuracy on a variety of computer vision tasks, including classification and recognition. The proposed system used MLPNN after preprocessing the image. The system accepts an input image of 16×16 and outputs one for the appropriate category and zeros for the rest.

The standard MLPNN framework consists of 3 layers: input, hidden, and output. The layers are comprised of nodes consisting of weights that either amplify or dampen the input data. The output of the layers will determine the classification of the image. The network consists of a forward and backpropagation algorithm to train the network. Backpropagation used the gradient descent optimization algorithm to adjust the weight of nodes by calculating the gradient of the loss function. The input layer served as the receiving end of the DWT/DCT combinational features and had nodes equal in dimension to the feature vectors. The output layer had nodes corresponding to the number of categories in the dataset.

The training samples to the MLPNN are represented by $\{p_1, n_1\}, \{p_2, n_2\}, \dots, \{p_Q, n_Q\}$, where $p_q = [p_{q1}, p_{q2}, \dots, p_{qR}]^T \in \Re$ are the input to the network. The input to the MLPNN is a two-dimensional matrix that represents each of the training image samples for each category. If the size of the images are $m \times m$ and there are p categories holding q samples each, then the input matrix will be $m \star m \times p \star q$.

The forward propagation uses weight initialization, calculation of the activation unit, weight adjustment, weight adaptation, and convergence testing. Initially, all weights are set to small random values ranging between -0.25 and 0.25 . Assume w_{ji} represents the weight between the j th layer

and i th layer where w_{j0} is the initial weight. The activation units are calculated sequentially, starting from the first layer. The activation function for each layer is calculated as follows in equation 3.32:

$$a_j = S_{yj} \left(\sum_{i=1}^I w_{ji} x_i - w_{j0} \right) \quad (3.32)$$

where a_j is the activation of the j th hidden unit for the pattern. x_i is the input to the layer; I is the number of nodes in the layer, and S is the activation function. In the proposed three-layer system, the sigmoid function was used as the activation function in the hidden and output layer.

At the output, the loss function is determined, which is the sum of squares of the absolute errors between the actual output and the desired output. The backward propagation uses differentiation to optimize the weights, which recursively propagated back from the output layer through the hidden layers to the input layer. To minimize the error E , the partial derivative of this error with respect to each weight in the network is computed. The change in the weights is proportional to the corresponding derivative, where η is the learning rate (which was set to 1.10 in the proposed system). The last term in the function is the previous weight change. At each step, n , the weights are updated. The equation is shown in 3.33.

$$\Delta w_{ji}[n+1] = -\eta \frac{\partial E}{\partial w_{ji}} + \Delta w_{ji}[n] \quad (3.33)$$

Note that Δw_{ji} are weight adjustments. Through the process of gradient descent, the process repeats until the desired output is reached. The output of this network is a vector with a weight

for each of the categories and testing samples. Neuron bias is added to the layers of the MLPNN to offset the origin of the activation functions. Bias allows for rapid convergence in the training process. The bias was set to a random value between -0.25 and 0.25 . In the same manner, for the other weights, the bias weights were trained. The weights were updated in the hidden layers, as shown in equation 3.34.

$$w_{ji}[n + 1] = w_{ji}[n] + \Delta w_{ji}[n + 1] \quad (3.34)$$

3.3.2 Multilayer Sigmoid Neural Network (MLSNN) Classifier

The classification stage, which performs the final step of image classification to recognize individuals, consists of an MLSNN. These MLSNNs are accurate, faster than convolutional neural networks, and less storage-intensive, and have become a standard robust feature extraction method for medical classification [53]. This approach has many of the benefits of neural networks, including the use of simple, homogeneous computations that are easy to parallelize on hardware. It also bears strong performance due to end-to-end training that formulates the problem as a single optimization solution, where all components of the model share the same end objective. The belief is that it will replace the most popular method of using the Euclidean distance for classification as in [54, 55, 56]. [57] has shown that using a simple neural network like the MLSNN provides a much better correlation between the input and the output than the popular Euclidean classification.

The MLSNN is a class of feedforward artificial neural network that is composed of layers of neurons (nodes) where the computations occur. The MLSNN used in this paper will have three layers: an input layer, a hidden layer, and an output layer. These nodes are activated when provided sufficient stimulus. This stimulus consists of input from the images, which is multiplied by a set of

weights and biases. The nodes amplify the inputs, which will better classify the image, and dampen those that will not. The network is trained using forward and backward propagation known as an epoch. In general, MLSNN is a specific type of MLPNN that uses sigmoid functions for the neurons. MLPNNs can have various numbers of layers, and these layers are densely connected. Multilayer sigmoid neural networks have been investigated for various applications. They were first used for pattern recognition [58] and then used to increase the quality of a human face image in [59]. More recently, MLSNNs were used in face gender recognition [60], automatic extraction of the eye and mouth fields from a face image [61], and human face detection in still images [62]. The work proposed here exploits the increasing quality advantage in the pattern recognition of the MLSNN. The work is different in that it recognizes a person from the images, and does not solely categorize female and male subjects or extract faces from a still image. To the best of our knowledge, we are one of the first to use feature extraction and MLSNN in combination with facial recognition.

3.3.3 Pretrained Convolutional Neural Networks

There are several pre-trained networks available on MATLAB R2018a that was used for testing convolutional neural networks. Installation of all support packages and toolboxes for each of the network was completed to support each network. The AlexNet, for example, was trained on a subset of the ImageNet database and is trained in more than a million images and can classify images into 1000 categories. In the algorithm, the pre-trained network loads all layers in the network except for the last three (referred to as transfer learning). Afterward, the network is trained using a new set of data that includes newly calculated weights for the last three layers.

3.3.3.1 AlexNet

This design was one of the primary deep learning systems to push ImageNet Classification accuracy by a considerable step in contrast with traditional methods. The neural network consists of 60 million parameters and 650,000 neurons. It is composed of five convolutional layers (some of which are trailed by max-pooling layers) and followed by three fully connected layers for a total of 8 learned layers.

A distinguishing factor of AlexNet, proposed by Krizhevsky [67], is that it uses ReLu(Rectified Linear Unit) to model a neuron's output f as a function of its input x as well as to help solve the vanishing gradient problem. When considering training time with gradient descent, ReLu is much more efficient. ReLu is given by $f(x) = \max(0, x)$.

On the test data of the ImageNet contest, the AlexNet system achieved a top-1 error rate of 37.5% and a top-5 error rate of 17.0%.

3.3.3.2 VGGNet

This architecture is from the Visual Geometry Group at the University of Oxford. It improves the performance of AlexNet by replacing large kernel-sized filters with consecutive 3x3 kernel-sized filters. Multiple stacked smaller size kernel generates better recognition rates than larger sized kernels. The multiple non-linear layers increase the depth of the network and, in return, learns more from complex features at a faster speed.

Three fully connected layers follow the VGG convolutional layers, as in the AlexNet network. The width of the first convolutional layers starts with a width of 64, and as the layers progress, the width increases by a factor of 2 after every pooling layer. On the test data of the ImageNet contest,

the VGGNet system achieved the top-5 accuracy rate of 92.3%.

VGG-16 and VGG-19 are two variants of the VGGNet. VGG-16 has a total of 16 layers consisting of convolution layers (3×3), max-pooling layers (2×2), and fully connected layers. The VGG-19 has a total of 19 layers comprised of convolution, max pooling, and fully connected layers.

3.3.3.3 *GoogleNet*

While VGG accomplishes an incredible precision on the ImageNet dataset, its utilization on even the most modest-sized GPUs is an issue as a result of enormous computational requirements, both as far as memory and time. It is wasteful because of the substantial width of the convolutional layers.

For example, the composition of the 3rd convolutional stage of the VGGNet with 16-layers is $3 \times \text{conv3-256}$ layers. The first layer has 128 input planes and 256 output planes. The other two layers have 256 input planes and 256 output planes. The calculation for the number of parameters for the two last layers is $256 \times 3 \times 3 \times 256 = 590,080$. The vast amount of parameters would need a considerable amount of time to process.

A convolutional operation at one location with an output plane connected to every input plane is called a dense connection architecture. GoogLeNet works on alleviating the fact that activations in a deep learning network have a value of zero and, therefore, unnecessary or superfluous because of the relationship between them. In this way, the most proficient design of a deep network will have a sparse association between the activations, which suggests that every one of the 512 input planes will not have an association with all the 512 input planes. There are methods to prune out such associations, which would bring about a sparse weight/association. However, kernels for sparse matrix multiplication are not advanced for GPU bundles which render them to be considerably

slower than their dense complements.

Therefore GoogLeNet, proposed by Simonyan [68], implemented a module called the inception module that approximates a sparse CNN with standard dense construction. Since just a few neurons are successful, the width/number of the convolutional filters of a specific kernel size is kept small. Additionally, it utilizes convolutions of various sizes to catch subtle details at a variety of scales that is 5×5 , 3×3 , and 1×1 .

Another notable point about the GoogleNet module is that it has a bottleneck layer, which is the 1×1 convolution layer. This layer helps in an enormous decrease in the calculation requirement.

Another change that GoogleNet made was to replace the completely connected layers toward the end with a straightforward global average pooling, which averages the channel values over the 2D feature map, after the last convolutional layer. This average pooling decreases the cumulative number of parameters, where fully connected layers contain approximately 90% of parameters. Utilization of a large system width and depth enables GoogleNet to expel the fully connected layers without influencing the accuracy.

GoogleNet accomplished a top-5 accuracy rate of 93.3% on ImageNet. Although VggNet achieves a phenomenal accuracy on the ImageNet dataset, its deployment on even the most modest-sized Graphics Processing Units (GPUs) is a problem because of substantial computational requirements, both regarding memory and time. It becomes inefficient due to the large width of convolutional layers, although it has been proven to be significantly quicker than VGG [107].

3.3.3.4 *Inception-v3*

GoogleNet proposed a deep convolutional neural network architecture which they codenamed Inception. For all intended purposes, this is Inception-v1. This version includes some sparsity in

the network as well as improved convergence in the classification error of intermediate layers. Szegedy in [108] built on the original version of Inception that is inception v1 and made some improvements. Some of the upgrades were decreasing the size of kernels that were larger than 3×3 , which led to more efficiency. Other enhancements included a factorization of convolutions and improved normalization. The final revision led to Inception-v3 that applied all of the improvements and surpassed its predecessor, GoogleNet, on the ImageNet database.

3.3.3.5 *ResNet*

There are two problems with the CNNs discussed earlier in this paper. The first is called vanishing gradient. Vanishing gradient occurs on CNN because it uses gradient-based learning methods and backpropagation. During each iteration of training of the network, the neural network's weights receive an update proportional to the partial derivative of the error function with respect to the current weight. The problem occurs when the gradient becomes vanishingly small and effectively prevents the weights from changing its value and virtually stops the training of the neural network.

The second issue is called the degradation problem. Degradation problems occur in training deeper networks and calculating the optimization of enormous parameter space, which leads to nonchalantly adding layers that prompt higher training error. Residual networks such as ResNet permit training of these deeper networks by building the system through modules called residual models.

ResNet, developed by He [109], has a 152 layer network (which was ten times deeper than previous CNN), which highlights unique skip connections and substantial utilization of batch normalization. Like GoogleNet, ResNet uses average pooling before the classification layer. It achieves better accuracy than VGGNet and GoogLeNet while being computationally more efficient than VGGNet [107].

3.4 Window Based Feature Extraction Algorithms

Feature extraction is crucial to the identification of the speaker as the input to the speaker model and pattern matching process. Two popular steps, framing, and windowing, are precursors to the feature extraction methods. Framing is the process where the speech signal is made stationary by dividing it into overlapping fixed duration segments called frames. Windowing is the process where each frame is multiplied by a window function that smoothes the effect of using a finite segment. In any speaker recognition system, it is essential to extract features from each frame that can capture the speaker-specific characteristics [110].

3.4.1 Real Cepstral Coefficients (RCC)

RCCs are computed by transforming the signal from the time domain to the frequency domain through a Fast Fourier Transform (FFT) for each frame. The log of the FFT of the signal and the application of the inverse Fourier transform (IFFT) returns the real cepstrum of the signal, and can be written as follows:

$$\text{Real Cepstrum} = IFFT(\log(FFT(s(n)))) \quad (3.35)$$

where $s(n)$ is the original windowed signal [111].

3.4.2 Mel Frequency Cepstral Coefficients (MFCC)

The human perception of the frequency content of sounds does not follow a linear scale. The Mel scale was derived and is roughly linear from 0 to 1 kHz and logarithmic above 1 kHz. In

MFCC, the signal follows the same process as the RCC algorithm. However, after the FT, the power spectrum (each frame), is first passed through a non-uniformly spaced (in frequency) bank of filters. Afterward, the log of the signal is computed. These non-linear frequency filters are called Mel frequencies and correspond to the logarithmic frequency distribution of the human ears' hearing. The logarithmic distribution in relations to actual frequency is shown below:

$$Mel(f) = 1000/\ln(1 + 10/7) * \ln(1 + f/700) \quad (3.36)$$

in this equation, f is in Hz.

3.4.3 Delta-Mel Frequency Cepstral Coefficients

The human ear is sensitive to both the dynamic and static characteristics of a signal, and the MFCC mainly reflects the static characteristics. The Δ MFCC (the first order of the MFCC), is appended to the MFCC to reflect the dynamic information of each frame. As a consequence, the combination is more robust [110]. To further improve the recognition rate, the second and third-order can be extracted to reflect more characteristics of the signal.

$$\Delta MFCC(m, n) = \frac{1}{\sqrt{\sum_{i=-k}^k i^2}} \sum_{i=-k}^k i \times MFCC(m, n + i) \quad (3.37)$$

3.4.4 Linear Prediction Coefficients (LPC)

LPC models the vocal tract by using an all-pole model, and the LPC features represent the main vocal tract resonance property in the acoustic spectrum. Linear Prediction coding estimates the

signal, $s(n)$, by a linear combination of the past p samples. For example,

$$s(n) = \sum_{k=1}^p a_k s(n-k) + e(n) \quad (3.38)$$

where $e(n)$ is the error term and the a_k^p values are referred as the linear prediction coefficients [112].

3.4.5 Linear Prediction Cepstral Coefficients (LPCC)

LPCC is computed through recursion from the LPC Parameters to the LPC cepstrum according to an all-pole model and exposes the differences of the biological structure of the human vocal tract [112]. The equation for the recursion is as follows:

$$c_n = \begin{cases} c_1 = a_1 & n < 1 \\ a_n + \sum_{k=1}^{n-1} \frac{k}{n} c_k a_{n-k} & 1 < n \leq p \\ \sum_{k=1}^{n-1} \frac{k}{n} c_k a_{n-k} & n > p \end{cases} \quad (3.39)$$

Islam [112] defines a_1, \dots, a_p as a p -order LPC feature vector, $c_n, n = 1, \dots, p$ as the coefficients, and p as the first p values of the cepstrum.

3.4.6 Perceptual Linear Predictive Cepstral Coefficients (PLPCC)

Perceptual Linear Predictive Cepstral Coefficients (PLPCC) are based on the magnitude spectrum of the speech analysis window. PLPCC is dissimilar to MFCC and LPC, which are cepstral methods, in that it is a temporal method and models the auditory speech spectrum through a low order

all-pole model. Revathi [113] details the steps followed to calculate the coefficients of the PLPCC. First, compute the power spectrum of a windowed speech. Second, group the results to 23 critical bands using bark scaling for a sampling frequency of 8 kHz. Third, perform loudness equalization and cube root compression to simulate the power law of hearing. Fourth, perform inverse Fast Fourier Transform (IFFT). Fifth, perform LP analysis by the Levinson-Durbin algorithm [114]. Lastly, convert LP coefficients into cepstral coefficients.

The relationship between frequency in Bark and frequency in Hz is specified as in

$$f(bark) = 6 * \operatorname{arcsinh}(f(Hz)/600) \quad (3.40)$$

3.4.7 *RelAtive SpecTrAl (Rasta) - PLPCC*

Rasta filtering takes the rate of change of nonlinguistic components in a speech that generally lies outside of the typical rate of change of the vocal-tract shape. This rate of change suppresses the spectral components that change either quicker or slower than the typical rate of change of speech. The RASTA approach can be combined with the PLPCC method to get the low-pass transfer function $H(z)$ [115]. RASTA filtering reduces the accuracy of the system in the absence of noise. However, it increases its accuracy significantly in the presence of severe noise [116].

CHAPTER 4: FACIAL RECOGNITION SYSTEM USING MIXED TRANSFORM AND MULTILAYER SIGMOID NEURAL NETWORK CLASSIFIER

4.1 Introduction

Facial recognition is a Biometric Artificial Intelligence-based technology that can recognize an individual by analyzing patterns found in their facial surfaces and shape [14]. This technology has received much attention in recent years because of its potential use in a vast array of applications in both law enforcement and civilian applications (e.g., government, mobile phone, social media, retailers, airlines, and marketers) [15].

Many existing face recognition methods utilize feature extraction in combination with Euclidean distance classification. This method often suffers from low recognition rates under poor lighting (for example, as found in the YALE dataset), excessive storage requirements, and slow recognition rate. Neural networks have recently become a potent tool for *image* recognition with high accuracy. The use of neural networks in *facial* recognition, however, has not been thoroughly examined. The overall strategy here is to improve on the feature extraction weaknesses, as well as use the efficiency of neural networks to create a system that is effective, quick, and accurate. The proposed method uses a combination of preprocessing, a Facial Mixed Transform (FMT) feature extraction, and a simple, and a high-speed classification through a MultiLayer Perceptron Neural Network (MLPNN). What makes our system novel is the use of a feature extraction stage before the MLPNN, as well as the use of an effectual MLPNN over the well-known Euclidean distance. More specifically, the MLPNN we use for classification is the MultiLayer Sigmoid Neural Network (MLSNN). The feature extraction stage uses a mixture of the two-Dimensional Discrete Wavelet

Transform (2D-DWT) and the two-Dimensional Discrete Cosine Transform (2D-DCT), which can extract the most relevant features of the facial images while minimizing detractors and reducing the dimensions of these features. Despite having only three non-convolutional layers, the MLSNN yields surprisingly good results, even when running directly on the preprocessed images without the feature extraction stage.

Many methods for feature extraction have been advanced in recent studies. [117] adaptively combines 2D-DWT coefficients and 2D-DCT coefficients recursively for image feature extraction and Euclidean distance classification. This method superimposes dominant coefficients from the two domains. The best results averaged over at least 462 trials for six training poses, obtained in [117], were 98.75%, 97.3%, 81.3% for the YALE, the ORL and the color FERET datasets, respectively. Gabor filters are applied in [118, 119] to extract the required features, followed by Principal Component Analysis (PCA), to minimize dimensions. [120] uses Linear Discriminant Analysis (LDA) and ADABOOST to find the best features of an image. The Hidden Conditional Random Fields (HCRF) model is also used for recognizing complex distributions via a mixture of Gaussian density functions [121]. [122] presented a summary of feature extraction approaches proposed by various authors and made conclusions on current trends in feature extraction techniques. The paper concluded that DCT and DWT techniques give good accuracy for facial recognition, but are not robust to image illumination changes. Gabor Filters, on the other hand, are robust to illumination changes, but produce redundant features and have problems dealing with high dimensions. PCA and Local Binary Patterns (LBP) reduce dimensions but do not guarantee accuracy.

In this work, the feature extraction stage applies two transforms, the 2D-DWT, and 2D-DCT, successively to determine the coefficients to represent the facial image optimally. The advantage of using both transforms is that the ideal coefficients can be effectively extracted in both domains due to the nonorthogonality of the coefficients in these domains. It has been observed that nonorthogonal basis functions in DWT lead to a complete and robust representational space that leads to better

performance over orthogonal basis functions[123]. That is, orthogonality is not a requirement for pattern recognition. The coefficient's importance is based on its position in the wavelength in the DWT domain and its energy level in the DCT domain. The resultant matrix leads to a minimization of noise and exclusion of image parts nonessential to the identification. The feature vector, which contains the final coefficients chosen from each domain, is then loaded into the MLSNN.

The popular Convolutional Neural Networks (CNNs), despite their accuracy, are typically considered computationally expensive, storage-intensive, and slow compared to feature extraction models in facial recognition[64][65]. However, by exchanging the CNN for an MLSNN, we can retain the classifying power of neural networks, while significantly reducing the required resources. The feature extraction stage can compensate for any losses due to the architectural simplification of the neural network. We find that the proposed system, which combines the feature extraction and MLSNN stages, outperforms both CNNs and other feature-extraction-based methods.

We make the following contributions in this paper:

1. In terms of recognition accuracy, the biorthogonal basis wavelet used by the 2D-DWT in the feature extraction model extracts the most relevant features of the facial image. It is ideal for this system because it filters noise efficiently and avoids introducing visual distortions.
2. In terms of decreasing the dimensions, the 2D-DCT is used in conjunction with the 2D-DWT in the feature extraction model. While many works have used the 2D-DWT and 2D-DCT for image recognition, the combination of these systems is novel in this work. Other algorithms, such as [117], have used mixed transforms adaptively, rather than successively as used in this paper. We also use the MLSNN instead of the simple Euclidean Distance Classifier (EDC).
3. In terms of processing time, the simple structure of the neural network classifier (MLSNN) allows for rapid training on the input datasets. Likewise, during the testing, the system will

identify the person associated with the facial image both accurately and quickly. This classifier contrasts with the slower and more complicated CNNs used in [65]. We note that the proposed system can be run efficiently without using Graphics Processing Unit (GPU) hardware, thus enabling potential use in embedded applications, for example. In the results, we showcase where the proposed system classifies quicker on a CPU than CNN-based systems running on GPU hardware.

4. In terms of recognition under illumination changes, for example, as found in the YALE dataset, the use of grayscaling in the system improves robustness.
5. In terms of the size of the dataset, several kinds of images from multiple datasets have been combined to better train the MLPNN for improved performance. To the best of our knowledge, no one else has combined these specific datasets.

Overall, the proposed model improves recognition and speed while reducing memory space requirements even when tested on a relatively large and diverse dataset.

The ORL, YALE, FERET, FEI databases were utilized to test the facial recognition system as well as a combination of all four datasets. In the results, we compare the recognition accuracy/processing speed of the proposed system (built using successive 2D-DWT and 2D-DCT with the MLSNN) against several facial recognition systems, a feature extraction system, and popular CNN systems. In the results, the proposed method overall maintained the recognition accuracy and substantially increased the speed of recognition, thus indicating that an appropriately chosen compact representation of the image makes for an ideal input to the MLSNN.

4.2 Proposed System

In this work, we create a model for recognizing individuals in a large dataset. The proposed system consists of a preprocessing stage, a feature extraction stage, and a classifier, as outlined in Fig. 4.1. In the diagram, we show an example of an actual image that was processed through the proposed system. For the preprocessing stage, all queried images were converted into grayscale before the feature extraction stage.

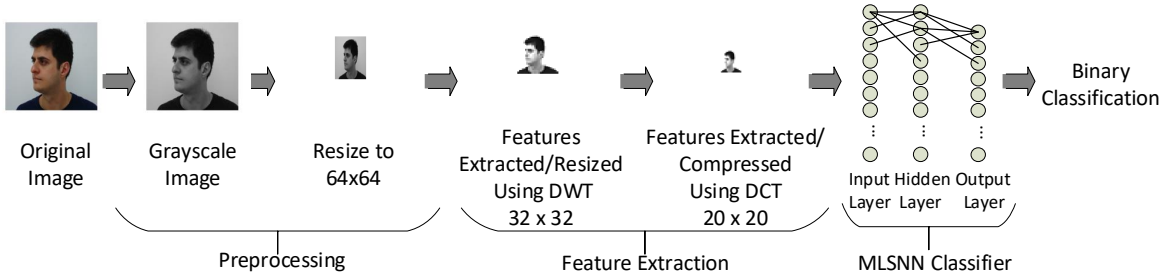


Figure 4.1: Block Diagram of Proposed System

4.2.1 Feature Extraction

The proposed feature extraction system is shown in Fig. 4.2. There are L possible nonorthogonal transformations depending on how many transforms are used in the system. For this reason, the algorithm performs at most L iterations, terminating early if the energy residual Φ , falls below 1%. For $l = 1, 2, \dots, L$, the transforms at the l^{th} stage are T_1, T_2, \dots, T_L . The transforms perform three tasks: transformation into another domain (i.e., frequency domain), extraction of features of an image through a weight matrix labeled C_1, C_2, \dots, C_L , and inverse transformation into the spatial domain to create the transformed coefficients, F_1, F_2, \dots, F_L . The transformed coefficients

are converted to a vector and kept to be concatenated at the end of the proposed system algorithm.

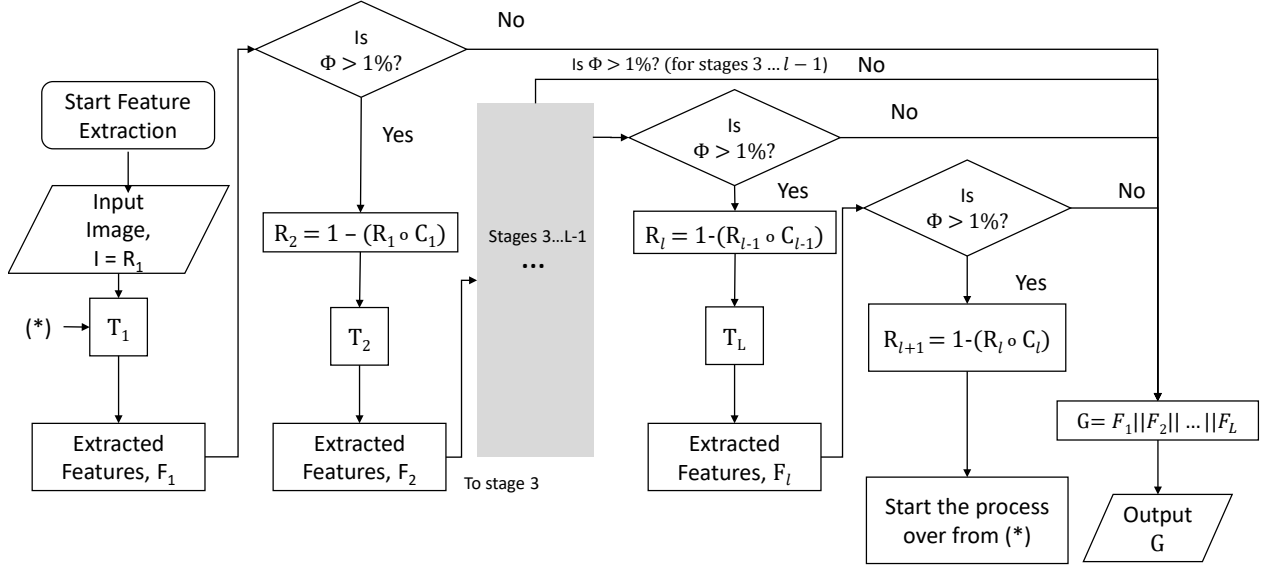


Figure 4.2: Block Diagram of Proposed Feature Extraction System

The residual is the result from $R_2 = 1 - (R_1 \circ C_1)$, where (\circ) is the Hadamard product. The energy residual (cost function) Φ is the difference between the input energy and the sum of the energies retained in each domain [117]. Specifically, it is defined as shown in Equation (4.1).

$$\Phi = P_I - (P_{F_L} + \dots + P_{F_2} - P_{F_1}) \quad (4.1)$$

The cost function, Φ , needs to be minimized. Through experimentation, the value of Φ that best represented the image and therefore gave the best recognition accuracy was determined to be 1%.

Consider the case where $F = 2$, using the two transforms 2D-DWT and 2D-DCT (which are nonorthogonal to each other). First, the image I (R_1) is transformed through the chosen transform

T_1 (in this case, the 2D-DWT), then its feature matrix is extracted through the approximate coefficients by using the BOW as the basis function. The extraction is achieved through a weight matrix C_1 , which is a matrix of ones and zeros defining the location of the coefficients in the image to keep. The result is F_1 . If Φ is greater than 1% then the residual R_2 is calculated by $R_2 = 1 - (R_1 \circ C_1)$ and this R_2 goes through the second transformation. The residual coefficients, R_2 , are resized to 32×32 . The BOW used in this research is the Bior2.2 wavelet. The extracted coefficients, F_2 , are resized to 32×32 . Next, the second transform T_2 , (in this case 2D-DCT), transforms the residual coefficients R_2 into the DCT domain, the feature matrix is reduced by taking only the coefficients that represent 99.99999% of the energy; in general, it was found that 32 coefficients captured this energy. The matrix C_2 is created to delineate the position of the coefficients to be kept, and the feature matrix is converted back to the spatial domain through the inverse 2D-DCT, and the resulting coefficients, F_2 , are resized to a dimension of 20×20 . For $L > 3$, we repeat the process, and for each l we terminate the process if $\Phi < 1\%$. After the algorithm terminates, each of the feature matrix F_l is concatenated to produce the final feature matrix G . The final feature matrix G , for each pose is converted into a one-dimensional vector to be loaded into the MLSNN.

4.2.2 Classification

In the classification stage, an MLSNN is invoked consisting of three layers: an input layer, a hidden layer, and an output layer. The input layer receives the image, the hidden layer approximates a continuous function, and the output layer makes a prediction about the input.

In the MLSNN, the activation function of a node defines the output of that node given an input or set of inputs. The activation functions are initialized using the values calculated from the feature extraction stage, x_i and are the input to the MLSNN, a_i are the calculated activation units, starting from the input layer as shown in equation (4.2). In equation (4.3) and (4.4), v_{ji} is the weight

between the j^{th} layer and the previous i^{th} layer and ω_{kj} is the weight between the k^{th} layer and the previous j^{th} layer. The activation function for the hidden and output layers are calculated using equations (4.2) - (4.4) (see [124]):

$$a_i = x_i, \quad (4.2)$$

$$y_j = S_{y_j} \left(\sum_{i=1}^I v_{ji} a_i + b_j \right), \quad (4.3)$$

$$z_k = S_{z_k} \left(\sum_{j=1}^J \omega_{kj} y_j + b_k \right), \quad (4.4)$$

where S_{y_j} and S_{z_k} are sigmoid activation functions, I is the total number of input nodes in the MLSNN, and J is the total number of individuals in the dataset. The algorithm updates the weights and biases so that the output from the network approximates $z(x)$ for all training inputs x . The network's weights and biases are updated by applying gradient descent using backpropagation. A cost function compares against a corresponding target value of y_d , which is the actual identity of the face image. The cross-entropy cost function C is

$$C = -\frac{1}{N} \sum_x [y_d \ln(z) + (1 - y_d) \ln(1 - z)], \quad (4.5)$$

where N is the total number of training images, the summation is over all training images, z is the vector of outputs from the network given input image x , and y_d is the actual classification, or label, of the facial image. Note that the output z depends on x , v , ω and b .

It is also important to note that the cross-entropy loss function was used instead of the more popular quadratic loss function to avoid the problem of learning slowdown [125]. The use of cross-entropy was appropriate in this case since loss measures the performance of a classification model whose output is a probability value between 0 and 1.

The backward propagation uses differentiation to optimize the weights, starting at the output layer and recursively to propagate through the hidden input layers. Minimizing the error E requires the calculation of the partial derivative of E with respect to each weight in the network, as shown in equation (4.6).

$$E = \frac{\partial C}{\partial \omega_j} = \frac{1}{N} \sum_x x_j (S(z) - y) \quad (4.6)$$

The error E was set to 0.01 because the recognition rate converged at this point for our system. The change in weight is proportional to the corresponding derivative, where η is the learning rate, which was set to 12.5 in our system). At each step n , the weights are updated. The equations are shown in (4.7) and (4.8).

$$\Delta v_{ji}[n+1] = \Delta v_{ji}[n] - \eta \sum_x \frac{\partial C}{\partial v_{ji}} \quad (4.7)$$

$$\Delta \omega_{kj}[n+1] = \Delta \omega_{kj}[n] - \eta \sum_x \frac{\partial C}{\partial \omega_{kj}} \quad (4.8)$$

Note that v_{ji} and ω_{kj} are weight adjustments. Throughout the process of backpropagation, the process repeats until the desired output is reached. The output of this network is a vector with a weight for each of the categories and testing samples. Neuron bias is added to the layers of the MLSNN to offset the origin of the activation functions. This neuron bias makes room for rapid convergence in the training process. The bias weights are trained in the same manner as the weights. The sigmoid neuron will be used in this research because of its versatility, and its function is continuously differentiable, which makes it relatively easy for backpropagation calculations.

The input layer serves as the receiving end of the extracted features from the FMT, and the number of nodes is set equal to the dimension of the feature vectors. The output layer produces one output value for each pose that belongs to an individual to be recognized. For this experiment, the number of epochs was dependent on the learning rate, which was set to 0.01. The training samples are represented by $p_1, n_1, p_2, n_2, \dots, p_Q, n_Q \in \mathbb{R}$, where $[p_1, p_2, p_3, \dots, p_Q]^T$ are the inputs to the network. p_q represents the intensity value of the q^{th} feature value in the face image. The input to the MLSNN is the set of feature traits obtained from the FMT for each of the training image samples. If the size of the features is $m \times m$, then the input vector will be of size $m * m$.

The first step of the forward propagation consists of the initialization of the activation functions, weights, and bias. The remaining steps are the calculation of the activation unit, weight adjustment, weight adaptation, and convergence testing. Initially, all weights and biases, v_{ji} , and b_i , are set to small random values ranging between -0.25 and 0.25 .

The proposed system consists of a training path and a testing path, as shown in Fig. 4.3. In all the datasets tested, the poses were divided with 80% used for training and 20% used for testing. For ten poses in a category, eight poses would be used for training, and two poses would be used for testing. The MLSNN is trained on a set of input facial images and their respective identities. The MLSNN learns to model the correlation between images and their identity. Training involves adjusting the weights and biases of the model in order to minimize errors. There is a continuous forward pass, and backpropagation pass called an epoch, where the equations in the backpropagation adjust the weights and biases relative to the error. We end training when the cross-entropy loss function reaches convergence; in our case, the value was 0.01.

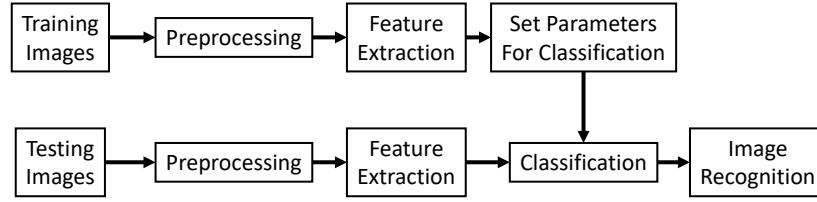


Figure 4.3: Proposed System with MLSNN

In the testing path, the identities of the input images are determined by passing the image through the same three stages described above and then classifying them via the weights and biases of the MLSNN. The testing stage starts by preprocessing the test pose with the same grayscale and resizing operations applied in the training stage. The resulting image is passed through the FMT to obtain its feature vector, which is then passed through the trained MLSNN to determine the identity of the facial image. The overall recognition rate is calculated by dividing the number of correctly identified poses by the total number of test poses.

4.3 Experimental Results

We first evaluate the proposed system separately on four databases: the ORL [126], YALE[29], FERET-c [127, 128], FEI [32]. Next, we test against a “combined” dataset built from all four databases to evaluate the system using a larger, more varied set of individuals and faces. The proposed system is compared against six face recognition (FR) systems and six CNNs. Fig. 4.4 shows the composition of Systems 1 through 6, as well as the proposed system.

The FR systems are constructed as follows: System 1 uses no FMT and only MLSNN. Systems 2 and 3 perform feature extraction with FMT, followed by the MLSNN: System 2 uses only 2D-DWT

for the extraction, and System 3 only uses 2D-DCT. Systems 4 and 5 perform feature extraction with FMT, followed by EDC. System 4 uses only 2D-DWT for the extraction, and System 5 only uses 2D-DCT. System 6 uses both 2D-DWT and 2D-DCT with the EDC. The proposed system uses both 2D-DWT and 2D-DCT with the MLSNN. Six CNNs are used for comparison as well, that is the AlexNet [67], VGG 16 [68], VGG 19 [68], GoogleNet [129], ResNet 50[109], ResNet 101[109].

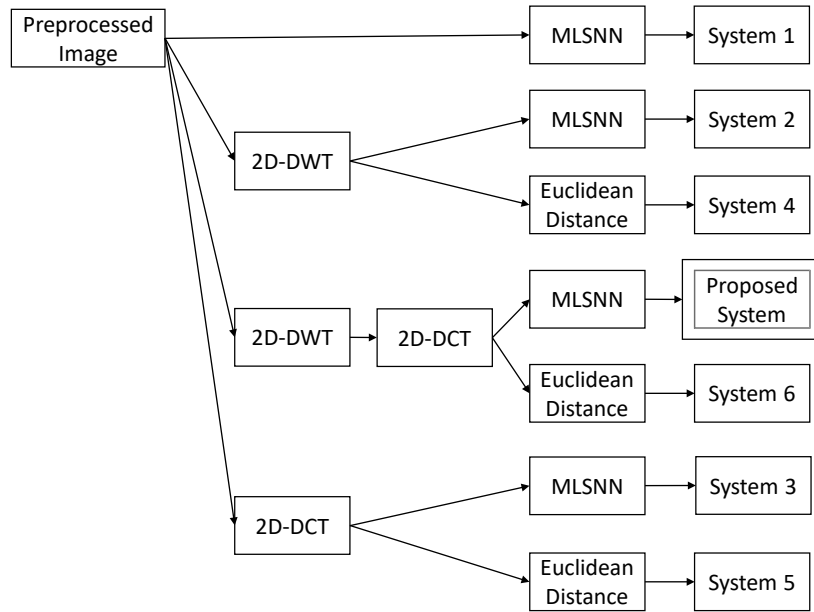


Figure 4.4: Training Stage of Systems 1 through 6.

The facial images used in this work were assembled from several different datasets. Specifically, 150 images were used from YALE, 400 images from ORL, 2000 images from FERET-c, and 2000 images from the FEI dataset. The YALE database contains 15 categories of individuals, with ten poses per individual. The original ORL and FERET-c dataset include 40 and 200 categories of individuals, respectively, with 11 poses per individual. We randomly chose ten of these poses

for both the training and testing datasets. The FEI dataset has 14 poses with 200 categories of individuals. The darkened images and the sideways images were removed to give ten poses per category. The “combined” dataset has a total of 4550 images of individual faces. There were a sizeable number of images such that the images could be divided into 280 categories containing ten poses each. The dataset was split with 80% of the facial images used for training and 20% used for testing.

The row sizes of the original face images vary from 112 pixels to 640 pixels, and the columns vary from 92 to 480 pixels. Since each of the original datasets uses a different image size, we automatically resized all images to a uniform size of 64×64 pixels. This size was found sufficient for capturing distinguishing features of the faces to enable identification. The images are resized using nearest-neighbor interpolation. After resizing the images to 64×64 , the wavelet transform can extract the significant features of the image, and at the same time, minimize the feature vector space of the image to a quarter of its size (32×32), and the discrete transform can further reduce the dimensions to 20×20 .

The simulations were tested on an Intel Core i7-8700K 3.7GHz six-core personal computer with an NVIDIA GTX 1080 Ti 11GB graphics card. The experiment used MATLAB 2018b for the simulations. The CNNs are pre-trained image classification networks available in the MATLAB Deep Learning Toolbox. We make use of transfer learning for the CNNs. Specifically, the pre-trained CNNs have already learned to extract significant and useful features from everyday images and can be used as a starting point to model our specific datasets. The majority of these networks are pre-trained on a subset of the ImageNet database [130], which is used in the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) [131]. These networks have been trained on more than a million images and can identify 1000 object categories, such as pool tables, coffee mugs, and various animals.

Results for all datasets are summarized in Tables 4.1 through Table 4.4. The final dimension size of the feature matrix to be classified is shown in Table 4.1 for each dataset. Recall that the “combined” dataset represents all four datasets combined. Table 4.2 compares the recognition accuracy of different methods on the five datasets tested. Table 4.3 compares the processing time per image in seconds for training and testing for all datasets and systems. Table 4.4 shows the testing processing time per image in seconds for all datasets.

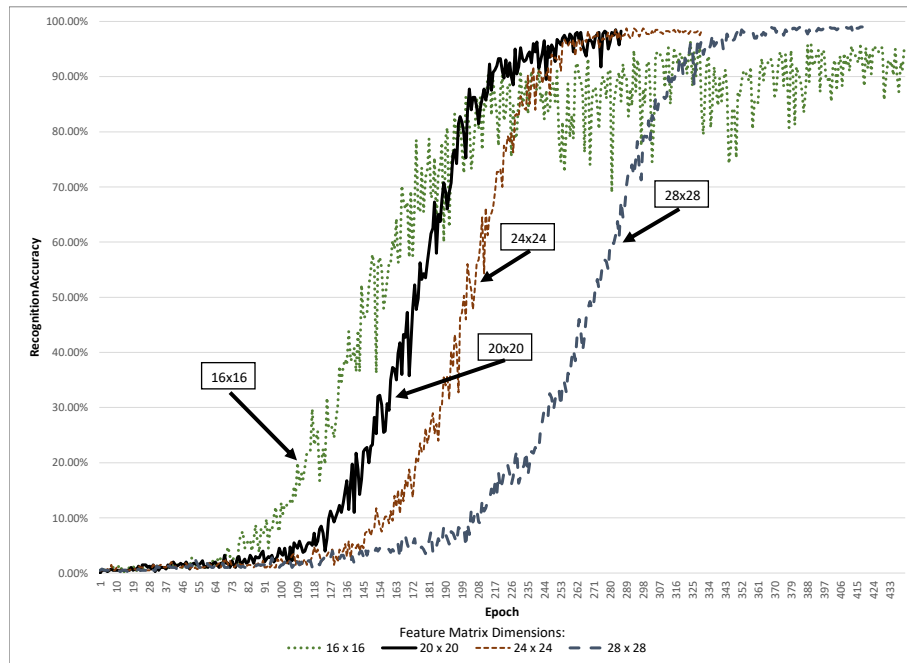


Figure 4.5: Recognition Accuracy for FEI Dataset Across Different DCT Compressed Sizes During Training of the System

Table 4.1: Image Dimension Comparison for the ORL, YALE, FERET-c, FEI and Combined Dataset and Proposed System for Classification

Database	Actual	CNN	Proposed
ORL	112×92	227×227	20×20
YALE	243×320	227×227	20×20
FERET-fc	384×256	227×227	20×20
FEI	640×480	227×227	20×20
Combined	Varies	227×227	20×20

Table 4.2: Recognition Accuracy (%) For All Datasets

	YALE	ORL	FERET-c	FEI	Combination
System 1	96.7	91.3	93.0	97.8	93.9
System 2	96.7	97.5	91.5	95.3	92.4
System 3	96.7	97.5	94.8	99.3	94.7
System 4	80.0	96.3	81.3	90.5	86.8
System 5	80.0	97.5	81.3	90.3	86.6
System 6	80.0	96.3	82.5	91.3	87.5
Alobaidi et.al [117]	86.7	92.5	77.3	87.7	97.3
AlexNet [67]	95.5	90.9	96.3	97.4	96.3
VGG 16 [68]	91.1	85.2	95.1	95.1	96.8
VGG 19 [68]	88.9	86.1	94.9	94.9	95.9
GoogleNet [129]	77.8	90.2	94.7	94.7	95.1
ResNet 50 [109]	80.0	80.3	94.7	94.7	96.7
ResNet 101 [109]	86.7	84.4	95.0	95.0	95.5
Proposed System	100.0	98.8	95.3	99.3	96.7

Table 4.3: Training + Testing Processing Time (s) Per Image for All Datasets

	YALE	ORL	FERET-c	FEI	Combination
System 1	25.83	610.31	3454.75	4501.25	7284.18
System 2	19.17	28.75	154.81	236.38	541.84
System 3	75.83	154.06	324.06	317.00	577.23
System 4	3.33	2.50	5.44	5.38	10.58
System 5	128.33	124.06	125.75	123.69	128.05
System 6	63.33	56.25	57.13	57.25	59.84
Alobaidi et.al [117]	1210.00	923.44	1299.88	6554.44	634.53
AlexNet [67]	84.17	66.56	204.50	1193.13	843.49
VGG 16 [68]	187.50	264.06	929.56	1695.75	1616.73
VGG 19 [68]	212.50	298.13	954.19	2603.94	2250.99
GoogleNet [129]	173.33	140.31	332.06	1355.69	887.47
ResNet 50 [109]	522.50	647.81	1527.31	2107.50	2344.59
ResNet 101 [109]	665.83	841.88	2111.00	3437.50	3556.43
Proposed System	833.33	308.75	59.56	62.06	26.57

Table 4.4: Testing Processing Time (s) per Image for ALL Datasets

	YALE	ORL	FERET-c	FEI	Combination
System 1	0.03	0.01	0.01	0.01	0.01
System 2	2.37	3.26	2.85	2.90	2.77
System 3	3.05	18.61	18.53	18.19	20.15
System 4	2.39	2.34	5.74	5.93	11.85
System 5	15.19	16.48	30.90	30.86	41.03
System 6	7.62	7.81	11.29	7.85	10.46
Alobaidi et.al [117]	3.64	7.04	36.29	35.41	322.26
AlexNet [67]	9.99	3.74	2.52	3.97	2.97
VGG 16 [68]	17.78	16.33	13.82	15.38	13.73
VGG 19 [68]	18.27	17.23	14.59	16.22	14.80
GoogleNet [129]	6.61	3.80	3.24	4.72	3.75
ResNet 50 [109]	10.72	8.19	7.33	8.97	8.79
ResNet 101 [109]	19.27	13.41	10.55	12.33	10.74
Proposed System	2.81	4.82	4.05	4.00	4.03

4.3.1 Remarks on the Results

In Table 1, the information shows that the proposed system will use an image with a much smaller size before it is processed and classified in the MLSNN. This smaller image will lead to fewer calculations and less processing time through the MLSNN. The time will be critical during the testing time for real-time applications and less critical during the training time, which is often done off-line. Table 3 shows the training and testing time for all datasets. It is important to note that the

proposed system sometimes has a higher processing time compared to other systems, especially for smaller datasets. This higher processing time is only a concern when the training time is of importance, which is generally not the case for most cases. The training stage's purpose is to train the weights of the MLSNN to be used in the testing during the actual use of the system. Table 4 shows that for the testing time, overall, the proposed system is faster than all systems except systems 1 and 2, where the accuracy is lower, as shown in Table 2.

In Fig. 4.5, we vary the feature matrix dimensions to search for the best 2D-DCT compression size. It was determined that the matrix dimensions that best represented the image were 20×20 . This dimension gave a less erratic loss function during training than the 16×16 dimension and allowed convergence to a high recognition accuracy at a faster rate.

The outcome of the experiment for system recognition demonstrates that the proposed system not only maintains the recognition accuracy compared to other popular methods but, in some cases, exceeds it. For the larger datasets such as the FEI and the “combined” dataset, the recognition rate is at par or surpasses those of the other systems. Overall, the accuracy of the proposed system is 95.3% or higher.

The outcome of the experiment for system processing time for testing + training reveals that the proposed system is faster than all of the CNN and is only slightly slower than the alternate systems tested. For the testing processing time, the proposed system runs competitively to the compared systems. It is important to note that the CNN algorithms run mostly on the GPU, and the proposed system runs on a CPU. Despite the parallel computing power of the GPU, which processed the CNN algorithms, the proposed system that operated on CPU alone indicated competitive processing time. Overall, on a system that consists solely of a CPU, the proposed system runs effectively.

Because the proposed system retains fewer coefficients per pose, the computational complexity of the proposed technique, especially in the testing stage, is much lower compared to the other

systems, especially the CNNs. Among the five datasets used, the proposed technique reduces the processing time by 30 – 97% compared with the systems under comparison.

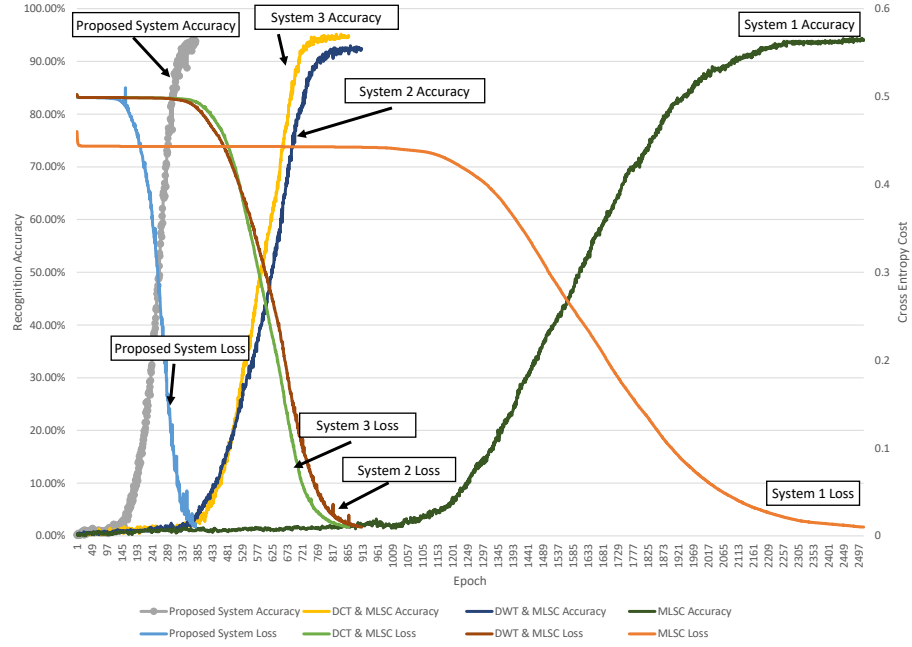


Figure 4.6: Recognition Accuracy and Cross-Entropy Loss of Combined Dataset for the Proposed System

In Fig. 4.6 all the systems that used the MLSNN as the classifier is compared. This figure shows the recognition accuracy and cross-entropy loss for the combined dataset. This diagram shows that the proposed system converges the fastest, i.e., the cross-entropy loss reaches the threshold of 0.01 at a slightly higher accuracy compared to the other systems.

4.4 Conclusion

A facial recognition system was proposed utilizing grayscaling and resizing, along with a mix of two popular transforms: the 2D-DWT and 2D-DCT. The FMT enhanced and compressed the image into a set of features, which better represented the image. These feature vectors were loaded into an MLSNN for the final classification stage, where results showed a very accurate classification. The smaller image meant that fewer parameters were required in the classifier, and ultimately decreased the processing time of the recognition. This method was trained and tested with a broad set of individuals. Overall, the proposed system performed very well in the larger dataset as compared to existing systems.

CHAPTER 5: DESIGNING CONVOLUTIONAL NEURAL NETWORKS FOR VARIOUS IMAGE RECOGNITION USING NEUROEVOLUTION

5.1 Introduction

Convolutional neural networks (CNNs) have received significant attention in recent years, in large part due to their outstanding behavior in complex supervised learning tasks [63]. These neural networks have had an especially large impact on computer vision applications such as facial recognition [64]. However, despite their accuracy, popular CNNs are typically computationally expensive, storage-intensive, and slow as compared to feature extraction models [65]. One of the most appealing ways to resolve these issues is to simplify the CNN architecture as much as possible while still maintaining high accuracy. This is the approach that we will follow. In general, recognition systems consist of three main stages: preprocessing, feature extraction, and classification. As with most CNN-based approaches, the CNN encompasses both the feature extraction and classification steps. We now describe the components making up our approach.

5.1.1 Convolutional Neural Networks (CNN)

CNNs were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of a human visual cortex [132, 133]. CNNs are customized versions of multilayer perceptrons (MLP), which are fully connected networks where each neuron in one layer is connected to all neurons in the next layer. Most CNNs that analyze images require a multitude of nodes and layers, making them complicated in terms of calculations and requiring significant processing power and time. The typical goal when building convolutional neural networks (ConvNets) is to design a deeper network that requires fewer parameters [66].

Existing CNNs and other neural network variants have obtained state-of-the-art accuracy in image classification [67], [68], [73], [134], facial recognition [69], [64], [70] and traffic sign recognition [71], [72]. Nonetheless, the depth of these architectures and the large number of parameters, require an enormous number calculations and computations, and large amounts of storage resources. These requirements restrict the usage of CNNs. Fortunately, deep models are considerably redundant in weights, filters, channels, and even layers [73]. By easing these demands, CNNs could be applied to increasingly broad applications: for instance, they could be used in smart cell phones, which have constrained computational power and memory capacity.

5.1.2 Feature Extraction

Feature extraction is a vital step in facial expression recognition [135]. Specifically, feature extraction is a process of dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for processing [136]. Traditional applications of feature extraction include [137, 138] in face recognition, and [139, 140, 141] in traffic sign detection and recognition.

Many methods for feature extraction have been advanced in recent studies. Alobaidi et al. in [117] adaptively combine two-dimensional discrete wavelet transform (2D-DWT) coefficients and two-dimensional discrete cosine transform (2D-DCT) coefficients recursively for image feature extraction and Euclidean distance classification. This method superimposes dominant coefficients from the two domains. The best results averaged over at least 462 trials for six training poses, obtained in [117], were 98.75%, 97.3%, 81.3% for the YALE, the ORL and the color FERET datasets, respectively.

In this paper, we focus on the creation of a compact CNN for feature extraction of images in faces and traffic signs datasets. Examples, where convolutional neural networks extract features from images in a dataset, include [142, 143, 144]. In this case, the convolutional filters represent the

characteristics (features) of the dataset images, and we typically seek to reduce the parameters of the CNN (and therefore reduce the required processing resources) without losing essential or relevant information. The best features to represent images in a dataset vary depending on the application. In facial recognition images, the best characteristics tend to be small facial details: eye separation, nose positioning, background color, and other minute attributes that compose a face. In traffic signs, on the other hand, the characteristics are more coarse: sign shape as exhibited by a stop sign (octagon) or a school bus sign (pentagon), the symbol or writing appearing on the sign (e.g., a diagram of school children crossing or the word “STOP”), and sign color.

A wide variety of approaches have been proposed for applying CNNs to the feature extraction step of facial recognition. [145] proposes a model that increases the number of images for a small number of employee images set of a small-scale company by applying different filters. This paper focuses on using CNNs to produce new artificially generate images to increase the size of a small dataset, which is different from our application. [146] proposes a cascaded noise-robust deep convolutional neural network (CNR-CNN) method, consisting of two sub-networks, i.e., a denoising sub-network and a face recognition sub-network, for face recognition under noise. This paper focuses on a robust CNN that works in a noisy environment, which is very different from our primary goal. [147] looks at the effect of CNN parameters, namely kernel size and the number of filters on the classification accuracy of a CNN using the FER-2013 dataset. This method uses CNN and investigates the best parameters of CNN, but they are limited to kernel size and the number of filters, and there is only one dataset tested.

Methods to find the features for traffic sign recognition are varied as well. For example, Kong et al. in [148] proposes a light-weight traffic sign recognition (TSR) algorithm based on cascaded CNN. The paper’s focus is to cascade a CNN and not to find the best parameters of a CNN. Chen et al. in [71] investigate the feasibility of using CNN for the task of traffic sign detection from camera images. The paper focuses on two specific issues: the low interclass variation of traffic signs and

the small size of the traffic signs in images. This paper looks at the feasibility of using a CNN and not the best architecture for a CNN to recognize an image.

5.1.3 Neuroevolution (NE)

NE is a subfield within artificial intelligence (AI) and machine learning (ML) that evolves a neural network through the use of evolutionary algorithms [75]. Although NE languished after its introduction almost three decades ago, it is making a comeback. Prominent artificial intelligence labs and researchers are experimenting with it, a string of new successes have bolstered enthusiasm, and new opportunities for impact in deep learning (DL) are emerging [75]. Its application for designing CNN for various image recognition application is just starting to show promise and will be further investigated in this paper.

A genetic algorithm is a search heuristic that is inspired by Charles Darwin’s theory of natural evolution. This algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation [149]. For complex problems, they usually perform better than traditional optimization methods, due to their capability of escaping local optima in the search space [104].

When building a CNN that best represents a dataset, a machine learning algorithm would need to exhaustively explore the parameter space in all the layers composing a convolutional network. This proves infeasible on even small neural networks. Backpropagation (gradient descent) is typically used to alleviate this problem. In contrast, several alternative solutions have been proposed in the NE community. Jalali et al. in [76] perform a neuroevolution method based on genetic algorithm for finding the optimal deep neural networks architecture in terms of hyperparameters, Badan et al. in [77] developed a NE method capable of evolving and optimizing CNNs with respect to the classification error and CNN complexity to reduce power consumption further. Zhang et al. [78]

developed a learning strategy based on NE to design and train optical neural networks. Aly et al. in [79] presented an evolutionary metaheuristic search to optimize deep neural networks and train a CNN, and Baldominos et al. [63] explored the application of NE to the automatic design of CNN topologies.

These NE methods use CNNs in the facial/traffic sign recognition, but they have a very different focus from our work. The simple CNN that we will build in this research are found using NE for eight datasets (as opposed to one dataset) and are tested across two different applications (facial and traffic sign recognition). Also, we will not be using the NE as the feature extractor but as the method to find the best CNN to extract the features of the images in the datasets.

Backpropagation-based DL used in CNN and NE have different qualities and shortcomings. DL is good at extracting structure from large amounts of data and producing a compact internal representation of a high-dimensional input. That is, a deep neural network can learn the characteristic features from a wide variety of pictures, from a baseball cap to a bird. However, DL performs poorly when solving problems with sparse rewards and simultaneously exploring many different strategies for solving a problem. NE can overcome these limitations. The combination of NE and DL will allow for the DL to make predictions or recognize objects based on a large number of examples and train a small action-selection component using NE with the pre-trained deep neural network as a back-end.

A system is built in two parts; there is a training module and a testing module. The training module is offline, and it is reasonable for extended processing time to determine the best parameters of a system. The testing accomplished in real-time, and it is where the image is passed through the already built system (CNN), and the image is recognized.

The structure of a CNN-based neural framework is complex and includes an enormous number of parameters that determine the effectiveness of the system to recognize and classify an image.

In this research, we endeavor to simplify this task, establish a technique that is prepared to generate a simplified total structure of a convolutional neural system explicitly produced to tackle three particular datasets: image dataset, facial dataset, and traffic sign dataset. This new structure will advance numerous parts of the architecture, which includes the number of layers, activation functions, batch size, and learning rate.

5.2 Proposed System

5.2.1 *Preprocessing*

Although a CNN system has preprocessing built-in, to keep the images consistent and all the same size, grayscaling and image resizing was done on the images. The importance of preprocessing is that it leads to better classification performances [150] and is paramount to increasing the speed of training such as centering and scaling techniques [151].

5.2.2 *Convolutional Neural Network (CNN)*

Convolutional neural networks achieve state-of-the-art accuracy on a variety of computer vision tasks, including classification, object localization, detection, recognition, and scene labeling [152]. The advantage of convolutional neural networks partially originates from their complexity (many parameters), which results in very high accuracy. An example of a large dataset is the noteworthy ImageNet dataset [153], which consists of 1000 categories with approximately 1000 image samples per category. However, creating a large labeled dataset is very time consuming and very costly. Also, the time it takes to process the image samples for training, as well as the storage space required, can become a problem in a society where time and space are of importance. To overcome this problem, creating an algorithm that can build the most efficient and accurate CNN

with the least complexity in terms of parameters and architecture can lead to faster processing time, decrease storage requirement, and keep the accuracy high for recognition of the image.

5.2.2.1 Aspect of Optimization

In our proposed system, we have considered the following parameters to be manipulated by the genetic algorithm with the following possibilities:

1. General hyperparameters

- Number of samples per category (S):

$$S = \begin{cases} 10 & \text{Datasets: YALE, ORL, FEI, and FERET-fc} \\ 50 & \text{Datasets: BTS, GTSRB, TSRD, and MNIST} \end{cases} \quad (5.1)$$

- Image Dimension (I_{dim}):

$$I_{dim} = (\text{floor}(R_n * 2)) * 32 + 32 \quad (5.2)$$

where R_n is a random real number in the range $[0,1)$. This leads to two possible image dimensions of 32×32 and 64×64 .

- Batch size (B):

$$B = 5 \quad (5.3)$$

The batch size was kept constant at 5 because it fit well with the sample size used of 10 for facial images and 50 for traffic sign images.

- Learning rate (η):

$$\eta = \left(10^{-(\text{floor}(R_n * 4) * 1 + 2)}\right) * (\text{floor}(R_n * 2) * 4 + 1) \quad (5.4)$$

where R_n is a random number in the range [0,1). This leads to the following values of 0.00001, 0.00005, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05.

2. Convolutional layers

- Number of convolutional layers (n_c):

$$\text{layers} = \text{floor}(R_n * 3) + 1 \quad (5.5)$$

This leads to one, two or three layers.

- Kernel size in each convolutional layer (k_{ci}):

$$k_{c1} = (\text{floor}(R_n * \chi_1)) * 2 + 1 \quad (5.6)$$

$$\chi_1 = \text{ceil}\left(\frac{\text{floor}\left(\frac{I_{dim}-1}{2}\right)}{2}\right) \quad (5.7)$$

Where χ_1 is calculated for a kernel size dimension from 1×1 to half the size of the I_{dim} , that is $\frac{I_{dim}}{2} \times \frac{I_{dim}}{2}$. The dimensions for the kernel size for the second layer and third layer are as follows:

$$k_{c2} = (\text{floor}(R_n * \chi_2)) * 2 + 1 \quad (5.8)$$

$$k_{c3} = (\text{floor}(R_n * \chi_3)) * 2 + 1$$

Where the new image dimension is calculated as

$$\begin{aligned} I_{dim2} &= \frac{(I_{dim} - k_{c1} + 1)}{2} \\ I_{dim3} &= \frac{(I_{dim2} - k_{c2} + 1)}{2} \end{aligned} \quad (5.9)$$

and χ_2 is calculated as follows

$$\begin{aligned} \chi_2 &= \text{ceil} \left(\frac{\text{floor} \left(\frac{I_{dim2}-1}{2} \right)}{2} \right) \\ \chi_3 &= \text{ceil} \left(\frac{\text{floor} \left(\frac{I_{dim3}-1}{2} \right)}{2} \right) \end{aligned} \quad (5.10)$$

- Number of nodes for each convolutional layer (d_{ci}):

$$\left\{ \begin{aligned} d_{c1} &= (\text{floor}(R_n * 6)) * 10 + 10 & n_c &= 1 \\ d_{c1} &= (\text{floor}(R_n * 6)) * 4 + 8 & n_c &= 2 \\ d_{c2} &= (\text{floor}(R_n * 6)) * 4 + 8 & n_c &= 2 \\ d_{c1} &= (\text{floor}(R_n * 6)) * 1 + 4 & n_c &= 3 \\ d_{c2} &= (\text{floor}(R_n * 6)) * 1 + 4 & n_c &= 3 \\ d_{c3} &= (\text{floor}(R_n * 6)) * 1 + 4 & n_c &= 3 \end{aligned} \right. \quad (5.11)$$

For one layer, the number of possible nodes is 10, 20, 30, 40, 50 and 60. For two layers the possible number of nodes for each layers is 8, 12, 16, 20, 24 and 28. For three layers the possible number of nodes for each layer is 4, 5, 6, 7, 8, and 9.

- Activation function in each convolutional layer (a_{ci}):

$$\epsilon = (\text{floor}(R_n * 3)) * 1 + 1; \quad (5.12)$$

The activation layer can be one of 3 possibilities:

$$a_{ci} = \begin{cases} \text{ReLU} & \epsilon = 1 \\ \text{tanh} & \epsilon = 2 \\ \text{sigm} & \epsilon = 3 \end{cases} \quad (5.13)$$

3. Fully Connected layers

- Number of fully connected layers (n_f):

$$n_f = \text{floor}(R_n * 2) + 1; \quad (5.14)$$

- Number of nodes of each fully connected layers (d_{ni}):

$$d_{ni} = (\text{floor}(R_n * 3)) * 32 + 32 \quad (5.15)$$

- Activation function in each fully connected layers (a_n):

$$\epsilon = (\text{floor}(\text{rand}(1) * 3)) * 1 + 1 \quad (5.16)$$

where

$$a_n = \begin{cases} \text{ReLU} & \epsilon = 1 \\ \tanh & \epsilon = 2 \\ \text{sigm} & \epsilon = 3 \end{cases} \quad (5.17)$$

These parameters are shown graphically in Fig. 3.7. It is important to note that the samples for the face recognition dataset were set to 10 because these datasets had at most ten poses per individual. Also, in the design, there is a pooling layer after each of the convolutional layers. Therefore the number of pooling layers will match that of the number of convolutional layers. The kernel size for pooling was taken as 2×2 , and therefore the height and width will be halved, and the overall size of the matrix will be quartered. The subsample method used was the average of the four cells of the matrix, and there was no overlap in the four-cell being averaged across the entire matrix.

5.2.3 Neuroevolution (NE)

In a NE system, the EA starts with an initial population; the *population* goes through the process of natural selection, which is the selection of two fittest individuals from the *population*. The two fittest individuals produce *offspring*, which inherit the characteristics of the parents, and if it is more fit than either parent, it will be added to the next generation. If parents have a better fitness score, then their offspring, the offspring will not be added to the population. Parents will then create another offspring that has a better fitness calculation. This process keeps on iterating, and in the end, a generation with the fittest individuals will be found. The EA process follows the algorithm shown in Algorithm 2.

Algorithm 2: Outline of Genetic Algorithm

- 1: Create a random initial *population*.
 - 2: Create a sequence of new populations - At each step, the algorithm uses the individuals in the current generation to create the next *population*. To create a new *population*, the algorithm performs the following steps:
 - a Calculate Raw Fitness Score: Score each member of the current population by computing its fitness value (the recognition accuracy).
 - b Select two individuals from the population-based on their fitness. They become the *parents*.
 - c Produce offspring from the parents: Children are produced by crossover (swapping the genes of each parent at a crossover point) and 0.015% parents will both mutate (one of the genes will be randomly chosen and its value will be changed to another random possibility).
 - d The current population is replaced with the offspring with the children to form the next generation.
 - 3: The algorithm stops when one of the stopping criteria is met.
-

In summary, there are five phases in an evolutionary algorithm.

1. Initial population
2. Fitness function
3. Selection
4. Crossover
5. Mutation

5.2.3.1 Initial Population

The generational process begins with a set of *individuals* consisting of *chromosomes* called a *population*. The chromosomes consist of a set of parameters (CNN parameters) known as *genes*. An individual is portrayed by a set of parameters known as genes. The *genes* will be the composition of the CNN, such as the number of layers and kernel size as detailed in the CNN section. Each *individual* holds in their genes a solution to the problem to be solved. In this case, the architecture, hyperparameters, activation functions needed to give the highest accuracy for the image to be recognized. In this research, the initial population was set to 50 *individuals*. The *genes* in the text chromosome were randomly determined as per the equation detailed in the CNN section.

5.2.3.2 Fitness Function

The fitness function determines the fitness score of an individual. In this research, the fitness score is recognition accuracy. The recognition accuracy was determined by the CNN built from the composition of *genes* in the text individual's *chromosome*. An individual competes with other individuals in the population based on the fitness score. The likelihood that an individual will be chosen to be a parent and produce an offspring depends upon the fitness score.

5.2.3.3 Selection

The selection phase chooses the fittest individuals and lets them pass their genes to the next generation. Two sets of *individuals* (textitparents) are chosen based on their fitness scores. The *individuals* reproduce through crossover and 0.015% of the time through mutation.

5.2.3.4 Crossover

Crossover is a significant phase in the genetic algorithm because this is the section in the algorithm where the best set of genes is determined to find a solution to the problem. That is, find the best set of genes to give the highest recognition accuracy. The *parents* are mated by exchanging *genes* at a crossover point. The crossover point is chosen at random from within the genes. The *offspring* is created by exchanging the genes of parents among themselves until the crossover point is reached. The new *offspring*, with its new genes, calculates its fitness score. The offspring are added to the population by replacing the lowest fitness score *individual* in the *population*.

5.2.3.5 Mutation

A mutation occurs to maintain diversity within the population and prevent premature convergence. New *offsprings* are also formed through a mutation of 0.015% of the time. A random *gene* in both parents is selected and randomly mutated. The fitness score of both *offsprings* is calculated, and the *offspring* replaces the lowest fitness score *individual* in the *population*. The mutation occurs 0.015%, and the offspring is a mutated version of both parents where one gene is chosen at random, and a new random characteristic is devised for that gene.

5.2.3.6 Termination

The algorithm terminates when the population converges, that is, the *offsprings* are not significantly different from the previous generation. At termination, the goal of finding the solution to the problem has been determined by the genetic algorithm.

Input				Convolutional Layers				Fully Connected Layers		
S	l_{dim}	B	η	n_c	k_{ci}	d_{ci}	a_{ci}	n_f	d_{fi}	a_{fi}
				x 3				x 2		

Figure 5.1: Details of Chromosome in the GA for all Datasets

5.2.3.7 Comments

The population has a fixed size of 50%, and as new generations are formed, individuals with the lowest fitness scores are removed from the population to provide space for new offsprings. The algorithm repeats to produce individuals in each new generation, which are better than the previous generation.

The definition of the chromosome in the genetic algorithm for all datasets is shown in Fig. 5.1.

5.3 Experimental Results

We first evaluate the proposed system separately on three sets of datasets. The facial recognition datasets: ORL [154], YALE [29], FERET-fc [127, 128], FEI [32]. The traffic sign datasets: Belgium-TSC [33], GTSRB [155], TSRD [156] and then the handwritten digit dataset: MINIST [34].

The facial images used in this work were assembled from several different datasets. Specifically, 150 images were used from YALE, 400 images from ORL, 2000 images from FERET-fc, and 2000 images from the FEI dataset. The YALE database contains 15 categories of individuals, with ten poses per individual. The original ORL and FERET-fc dataset include 40 and 200 categories of

individuals, respectively, with 11 poses per individual. We randomly chose ten of these poses for both the training and testing datasets. The FEI dataset has 14 poses with 200 categories of individuals. The darkened images and the sideways images were removed to give ten poses per category. The dataset was split with 80% of the facial images used for training and 20% used for testing for the calculation of accuracy.

The Belgium-TSC dataset consists of images of traffic signs found in Belgium. The original dataset consisted of 62 categories with a variety of numbers of images per category. Some of the categories had less than 50 images, so image duplication was used to have at least 50 images per sample. Therefore, no more than 50 random samples were selected for this dataset. Please note that some categories had in upwards of 290 images. The GTSRB dataset has more than 40 categories and more than 50,000 images in total. Only 40 categories were used in the experiment because some of the categories had few samples. The TSRD dataset has 6164 traffic sign images containing 58 sign categories. The images are divided into two sub-databases as a training database and a testing database. The training database consists of 4170 images, while the testing one contains 1994 images. The images were combined for the training and testing, and since some categories had few images, the number of categories tested was decreased to 55. The MNIST Dataset consists of handwritten decimal digits from 0 to 9. The MNIST dataset consists of 10 categories with approximately 6000 images per category. Some of the categories had less than 6000 images, and due to time constraints so no more than 750 random sample was selected for this dataset. Note that the range of images is from 5421 to 6742 images per category.

The row sizes of the original face images vary from 112 pixels to 640 pixels, and the columns vary from 92 to 480 pixels. Since each of the original datasets uses a different image size, we automatically resized all images to a uniform size of either 32×32 or 64×64 pixels. This size was found sufficient for capturing distinguishing features of the images to enable identification. The images are resized using nearest-neighbor interpolation.

The simulations were tested on an Intel Core i7-8700K 3.7GHz six-core personal computer with an NVIDIA GTX 1080 Ti 11GB graphics card. The experiment used MATLAB 2018b for the simulations. The convolutional networks are pre-trained image classification networks available in the MATLAB Deep Learning Toolbox. We make use of transfer learning for the CNNs. Specifically, the pre-trained CNNs have already learned to extract significant and useful features from everyday images and can be used as a starting point to model our specific datasets. The majority of these networks are pre-trained on a subset of the ImageNet database [130], which is used in the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) [131]. These networks have been trained on more than a million images and can identify 1000 object categories, such as pool tables, coffee mugs, and various animals.

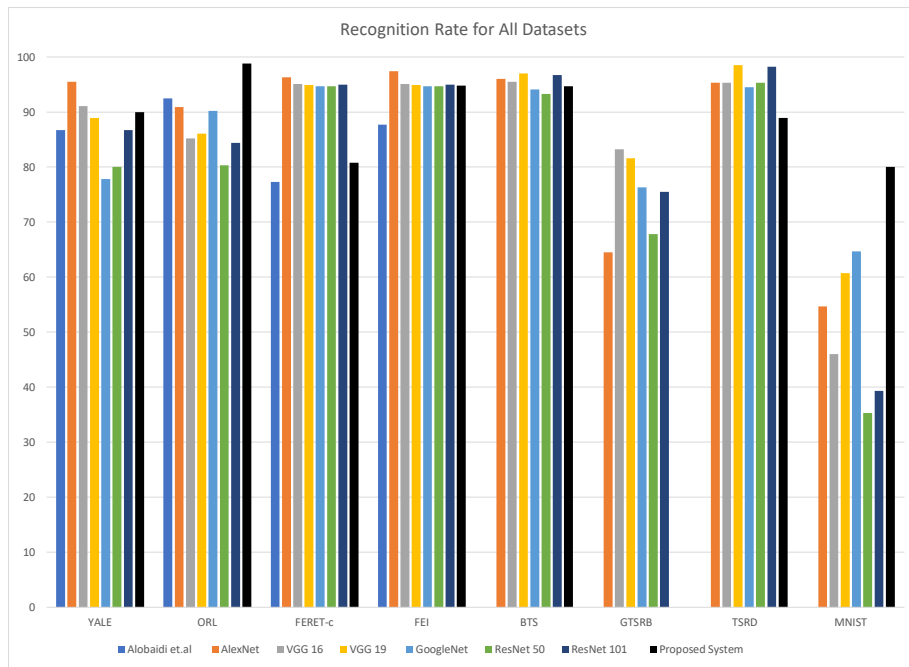


Figure 5.2: Recognition Accuracy for ALL Datasets

Table 5.1: Image Dimension Comparison for the ORL, YALE, FERET-fc, FEI and Combined Dataset and Proposed System for Classification

Database	Avg. Actual	CNN	Proposed
ORL	112×92	227×227	32×32
YALE	243×320	227×227	32×32
FERET-fc	384×256	227×227	32×32
FEI	640×480	227×227	64×64
BTS	126×114	227×227	32×32
GTSRB	384×52	52×227	32×32
TSRD	132×141	227×227	32×32
MNIST	28×28	227×227	32×32

Table 5.2: Best Parameters for All Datasets

	YALE	ORL	FERET-fc	FEI	BTS	GTSRB	TSRD	MNIST
Image Dimension	32	32	64	64	64	32	32	64
# of Samples	10	10	10	10	50	50	50	50
Batch Size	5	5	5	5	5	5	5	5
# Conv. Layers	1	1	1	1	1	1	1	1
Kernel Size Layer 1	15	5	11	5	23	11	7	27
Kernel Size Layer 2	-	-	-	-	-	-	-	-
Kernel Size Layer 3	-	-	-	-	-	-	-	-
# of Nodes Layer 1	50	40	50	30	30	40	60	50
# of Nodes Layer 2	-	-	-	-	-	-	-	-
# of Nodes Layer 3	-	-	-	-	-	-	-	-
AF for Conv. Layer 1	tanh	tanh	ReLU	sigm	sigm	ReLU	tanh	ReLU
AF for Conv. Layer 2	-	-	-	-	-	-	-	-
AF for Conv. Layer 3	-	-	-	-	-	-	-	-
# of FC Layers	2	1	1	1	1	2	1	2
# of Nodes FC Layer 1	96	-	-	-	-	64	-	64
AF for FC Layer 1	rect	-	-	-	-	tanh	-	sigm
AF for FC Layer 2	sigm	sigm	sigm	sigm	sigm	sigm	sigm	sigm
Learning Rate	0.005	0.01	0.005	0.05	0.01	0.005	0.01	0.01

Table 5.3: Recognition Accuracy (%) For All Datasets

	YALE	ORL	FERET-fc	FEI	BTS	GTSRB	TSRD	MNIST
Alobaidi et.al [117]	86.7	92.5	77.3	87.7	92.3	80.0	89.9	.0
AlexNet [67]	95.5	90.9	96.3	97.4	96	64.5	95.3	58.0
VGGNet 16 [68]	91.1	85.2	95.1	95.1	95.5	83.2	95.3	68.0
VGGNet 19 [68]	88.9	86.1	94.9	94.9	97	81.6	98.5	63.0
GoogleNet [129]	77.8	90.2	94.7	94.7	94.1	76.3	94.5	72.0
ResNet 50 [109]	80.0	80.3	94.7	94.7	93.3	67.8	95.3	41.0
ResNet 101 [109]	86.7	84.4	95.0	95.0	96.7	75.5	98.2	23.0
Proposed System	90.0	98.8	80.8	94.8	94.7	84.9	91.2	95.4

Table 5.4: Complexity Parameters for Popular CNNs

	AlexNet	VGGNet	GoogleNet	ResNet
Image Dimension	227	224	224	224
# Conv. Layers	5	16	21	151
Kernel Sizes	3,5,11	3	1,3,5,7	1,3,7
# of Nodes	96-384	64-512	64-384	64-2048
# of FC Layers	3	3	1	1
# of Nodes FC Layers	4096/4096/1000	4096/4096/1000	1000	1000
# of Parameters	61M [67]	138M [68]	6.8M [129]	25.6M [109]

Results for all datasets are summarized in Tables 5.1 through Table 5.4. The final dimension size of

the feature matrix to be classified, as shown in Table 1 for each dataset. Table 2 tabulates the results of the NE on the most simplistic CNN with the best accuracy for all datasets. Table 3 compares the recognition accuracy from the parameters chosen by the proposed system with [117] methods of mixed transform feature extraction and six popular CNN. Table 5.4 compares the complexity for popular CNNs.

5.3.1 *Remarks on the Results*

In Table 5.1, the actual two dimensional (grayscaled) average size of the images in the dataset, the resized two dimensional (grayscaled) image size used in CNN for all networks (except for AlexnNet which is (227×227)), and the proposed grayscale size of the image for the proposed system, is compared. The comparison shows that the traditional CNN requires much bigger sized image dimensions than the proposed system. It is a size decrease of 98%. This is important when it comes to decreasing computational complexity, computational speed, and storage requirements during computational training.

In table 5.2, the best parameters needed to compose the CNN found by the NE is shown. AF stands for activation function, and FC stands for fully-connected. It can be seen from the table that the number of convolutional layers that give the best recognition accuracy is one for all datasets. The kernel size varies dramatically between all datasets, as well as the number of nodes required for the first layer. The activation function also varies across the three possibilities of tanh, sigmoid, and ReLU. Three datasets prefer two FC layers, and the activation function has no favorites. The last FC layer is always the sigmoid function because it is conducive to the better cross-entropy loss function. The learning rate varies from 0.005 to 0.01 and 0.05, which is much fewer possibilities than the choices available. The learning rate of 0.01 is the most popular out of the dataset tested.

In table 5.3, it is important to know the number of input and output layers as this determines

the number of weights and biases that make up the parameters of the neural network. The more parameters in the network, the more parameters need to be trained, which results in longer training time. Training time is significant for deep learning as it a limiting factor unless there is access to powerful computing resources such as a computing cluster [157].

The number of parameters for the CNN is based on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) with 1000 categories. The network with the least amount of parameters is Googlenet, with only 6.8 million parameters. When compared to the number of parameters required for the proposed system, only the FEI dataset comes close to that number, and it still 1.4 million less than the GoogleNet algorithm.

In Table 5.4, the proposed system is compared to popular CNNs and current research from [117]. CNN using the parameters computed by the NE, always had competing results to that of the popular CNNs. This result shows that a simple CNN structure can be competitive with other more complex CNN structures. Compared to [117], the proposed system always had better results, which shows that a well designed CNN can do the work of feature extraction and classification without any extra manipulation.

5.4 Conclusion

In this paper, a neuroevolution technique was proposed for optimizing CNN architecture, hyper-parameters, and activation functions. This algorithm tunes the structure and weights of CNNs to reduce computational burden and memory requirements of the network during feature extraction, while still maintaining high accuracy. The effectiveness of the algorithm was demonstrated in two specific applications: facial images and traffic signs. The proposed neuroevolution algorithm was used to generate multiple CNNs, all of which contained a smaller total number of parameters as

compared to well-known CNNs such as AlexNet, VGGNet, ResNet, and GoogleNet. At the same time, the generated CNNs maintained high recognition accuracies, which were competitive with the well-known CNNs. The high accuracy of the proposed system indicates that there is often significant redundancy in existing CNNs for these particular applications, and also efficient feature extraction is possible with simpler topologies.

CHAPTER 6: ADAPTIVE FEATURE EXTRACTION ALGORITHM USING MIXED TRANSFORMS FOR FACIAL RECOGNITION

6.1 Introduction

Biometric face recognition technology has received significant attention in the past several years due to its potential in different applications [56]. Face recognition is the process of identifying a person from images or videos that extract unique facial features using various algorithms. Several face recognition approaches have been proposed [158, 159]. The feature extraction process is a very critical step in expressing facial images that could greatly affect the rate of recognition [160]¹.

Extracting a person's feature traits is the key to face recognition. Briefly described, here are two feature extractors prevalent in the literature. Discrete Wavelet Transform (DWT) is an implementation of the wavelet transform that decomposes an image into a mutually orthogonal set of wavelets [161]. Discrete Cosine Transform (DCT) is similar to the discrete Fourier transform in that it transforms an image from the spatial domain to the frequency domain. However, it represents an image as a sum of sinusoids of varying magnitudes and frequencies.

In [162], the preprocessing included Grayscale conversion, resizing, Laplacian of Gaussian Blur, Gamma Intensity Correction, Salt and Pepper noise detection, and Median and Weiner filters. The system applied two feature extraction algorithms (DWT and DCT) to extract features from the images. The truncation in the DCT domain utilized a new approach called Slope Triangular (STDCT). The Binary Particle Swarm Optimization algorithm is employed to remove outliers from

¹In this chapter, we partially use the material published in IEEE International Midwest Symposium on Circuits and Systems, 2018 [5].

the features. In [162], the maximum reported results, for an average of 25 experiments, were 72% for the color FERET-fc, 91.9% for the ORL, and 98.7% for the JAFFE database.

This paper presents a face identification system based on an adaptive algorithm that cascades the DWT and DCT extraction methods to extract the best features of an image. The Least Absolute Errors (LAE), also known as the $L1$ normalization, will perform a comparative evaluation of the feature extraction methods using images from three popular databases employed in facial recognition.

6.2 Proposed system

The proposed system is shown in Fig. 6.1 includes the training and testing phases of each image in the database.

All of the databases used in this paper have a certain amount of people with at least 10 poses per person. Each of the training/testing poses will go through two pathways. One of the pathways will be through the DWT, and the other pathway will be through the DCT.

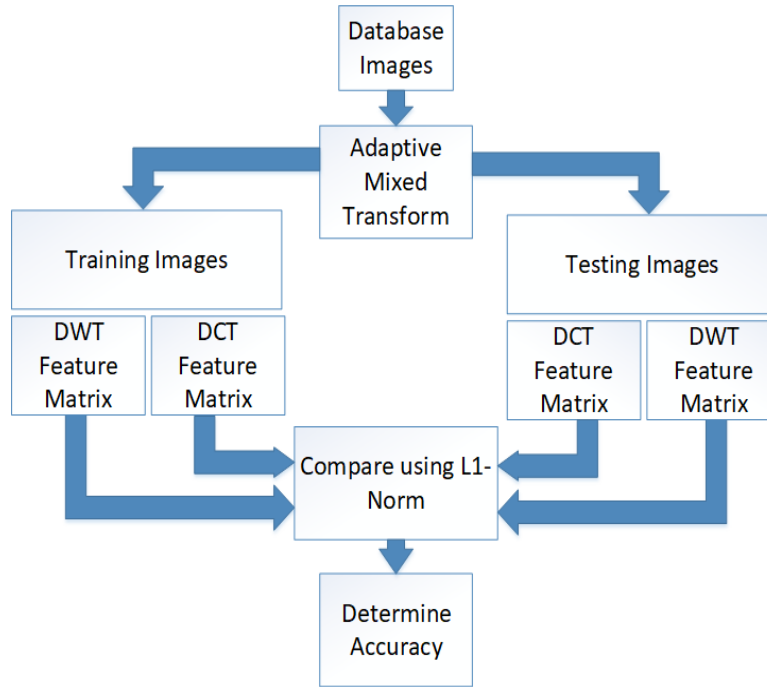


Figure 6.1: *Block Diagram of Proposed Facial Identification System*

For the DWT pathway, each of the poses is transformed using the DWT and the DCT; this DCT is part of the mixed facial identification system. This mixed system uses the DCT as a filter that follows the DWT and is recursively iterated until a threshold is reached. The threshold will be determined by comparing the energy change of the output of both the DCT and DWT to the energy of the original pose.

In the proposed system, each pose, A , is transformed first through the DWT, and then half of the

coefficients, C_0 , with the highest values are kept.

$$C_0 = \text{Max}(\text{dwt}(A)) \quad (6.1)$$

Each of these coefficients is then multiplied by a weight matrix W . The initial weight matrix $W_0 = [\alpha_0]$ which consists of $\alpha_0 = 0.8$ which is multiplied using the Hadamard Product, and then this product is transformed back into the spatial domain, D . The first iteration is shown in (6.2).

$$D_0(\alpha) = \text{idwt}(C_0 \circ W_0(\alpha)) \quad (6.2)$$

The DCT then transforms the original matrix A , and half of the coefficients, E_0 , with the highest values are kept.

$$E_0 = \text{Max}(\text{dct}(A)) \quad (6.3)$$

Each of these coefficients is then multiplied by a weight matrix of J . The initial weight matrix $J_0 = [\beta]$, which consists of an initial value of $\beta = 0.6$, which is multiplied using the Hadamard Product. This product is transformed back into the spatial domain, T_0 , becoming the residual of the image. This residual is then transformed by the weighted product of the DWT and the DCT. The first iteration is shown in (6.4).

$$T_0(\beta) = \text{idct}(E_0 \circ J_0(\beta)) \quad (6.4)$$

Next new α and new β are calculated based on the energy residual, $\Phi(\alpha, \beta)$ for each domain: Wavelet domain and Cosine domain. The energy residual is calculated as in (6.5) and following [163].

$$\Phi(\alpha, \beta) = [A]^2 - [D_n(\alpha)]^2 - [T_n(\beta)]^2 \quad (6.5)$$

Where n is the iteration index. For each iteration, the new α is calculated as the change in the residual energy over the original energy as shown in (6.6).

$$\alpha_n = \frac{\nabla_{\alpha_n} \Phi_n(\alpha, \beta)}{[A]^2} \quad (6.6)$$

Similarly, the β is calculated as shown in (6.7).

$$\beta_n = \frac{\nabla_{\beta_n} \Phi_n(\alpha, \beta)}{[A]^2} \quad (6.7)$$

The new weights are calculated as $W_n = [\alpha_n]$ and $J_n = [\beta_n]$.

The matrix T_0 is then transformed back into the DWT in the same manner as before to create D_1 eventually and then through the DCT as well to be transformed back into the spatial domain and become T_1 . As the image transforms into D_2 and T_2 continually until the value of residual over the original energy becomes less than 0.5%, which is the threshold.

When the iteration stops the weight matrices W_0, W_1, W_2, \dots are added to create a final weight matrix M (6.8).

$$M = \sum_{i=0}^P W_i \quad (6.8)$$

Where i counts the number of times the image was iterated through both the DWT and the DCT. P is the number of times the image was iterated. Note that the weights J is only used for filtering and is not used as part of the feature matrix.

The original discrete wavelet is multiplied by the final weight matrix, M , and this becomes the feature matrix of the DWT for the first pathway.

In the second pathway, the original pose, A , is transformed by the DCT and becomes the second feature matrix of the DCT.

This means that each training pose will consist of two pathway feature matrices.

6.3 Experimental Results

The experiment used three databases, the ORL [28], YALE [29], and FERET-fc [30, 31] sources for facial images.

The YALE database contains 150 face images of dimensions 243×320 for 15 individuals. There are 10 images per subject, one for each facial expression or configuration: center-light, glasses/no glasses, happy, normal, left-light, right-light, sad, sleepy, surprised, and wink [164]. The ORL database contains 40 individuals with 10 poses. Each pose has the dimensions of 112×92 . The

poses vary in position, rotation, scale, and expression as well as in each 10 samples; some have open/close eyes and are smiling/not smiling. The FERET-fc database contains many facial images in a variety of different conditions (e.g., facial expression, illumination, angle) [165]. In this paper, we regarded subset-Fc, which comprises of 200 people with 11 images per person with dimensions of 384×256 .

The experiment used MATLAB 2017b as the development environment and evaluated a face identification system that consisted of two feature extraction methods running in parallel and a classification system. There are four combinations of feature extraction methods compared in this experiment, which is labeled system1, system2, literature reference [162], and the proposed system.

This paper looked at two different variations of using DWT and DCT, as well as the proposed method. The proposed cropped dimensions for all the databases are 32×32 . The first variation is called System1 labeled in Tables 6.1 through 6.3. This system tested a truncated DWT with a truncated DCT in parallel. System2 consists of a truncated DWT with a predetermined number of maximum coefficients in the DCT, again in parallel. For all databases and systems, the dimensions of the final feature matrices are 24×24 for the truncated DWT, 6×6 for the truncated DCT, and 36 maximum coefficients for the DCT. The final feature matrix dimensions for the proposed system are 5×5 for the adaptive DWT and 6×6 for truncated DCT. Once both feature matrices are created for each pose, then the DWT feature matrix of the test image is compared to the DWT of the training images, and the DCT feature matrix of the training matrix is tested against the DWT of the testing image. The $L1$ normalization is used to find the distance between each pair. Next, the distance is normalized, and the smallest distance is determined to match the image. In Tables 6.1 through 6.3, the average recognition rate is the average of all combinations of training poses is that it is 4, 5 or 6. An example is that if there are ten poses and four training images, then all combinations are $\binom{10}{4}$, which equals 210 combinations to be tested and averaged. In the Tables,

the maximum recognition rate is the best recognition rate for all combinations.

Results of the experiments for recognition rate of the different feature extraction algorithms are summarized in Tables 6.1 through 6.3.

6.3.1 System Performance Evaluation using the ORL Database

For the ORL database, the proposed system outperformed System1, System2, and the literature [162] for six training poses for all combinations and was very near the same recognition rate for training poses 4 and 5. For the average, the proposed system outperformed System1, System2, and the literature [162] for all training poses.

Table 6.1: Maximum and Average Recognition Rates for ORL Database For All Combinations of Training Poses

Systems/Training Poses	Maximum			Average		
	4	5	6	4	5	6
System1	95.8	97	97.5	90.9	93.2	94.3
System2	95.8	97	97.5	90.9	93.2	94.3
[162]	89.6	97	98.1	89.6	92.0	93.6
Proposed	97.9	99	99.4	93.5	95.4	96.2

Table 6.2: Maximum and Average Recognition Rates for YALE Database For All Combinations of Training Poses

	Maximum			Average		
Systems/Training Poses	4	5	6	4	5	6
System1	97.8	98.7	98.3	83.4	84.3	84.9
System2	97.8	98.7	98.3	83.4	84.3	84.9
[162]	88.6	94.4	96.0	76.7	78.5	79.8
Proposed	93.3	97.3	100.0	81.7	83.0	84.1

Table 6.3: Maximum and Average Recognition Rates for FERET-fc Database For All Combinations of Training Poses

	Maximum			Average		
Systems/Training Poses	4	5	6	4	5	6
System1	74.1	79.1	84.6	60.2	64.9	68.5
System2	74.1	79.1	84.6	60.2	64.9	68.5
[162]	59.3	65.0	70.7	45.0	50.4	54.3
Proposed	83.6	86.9	90.0	72.0	74.8	76.8

6.3.2 System Performance Evaluation using the YALE Dataset

For the YALE dataset, the proposed system was only slightly lower than the System1 and System2 in both the maximum and average recognition rates. Only once was the proposed system better, and this was when the maximum recognition rate at six training poses had a recognition rate of

100%. The proposed system, however, outperformed the literature review [162] for all training poses for both the maximum and average recognition rate.

6.3.3 System Performance Evaluation using the FERET-fc Dataset

The FERET-fc data is much bigger than the two previous datasets, and for both the average and the maximum recognition rate, the proposed system was better than both system1, system2, and the literature review [9].

6.3.4 Remarks on the Performance of the Proposed System

Three design parameters need to be optimized: the recognition rate, the storage size, and the computational complexity. The proposed system improved the recognition rate compared to System1, System2, and the literature review, as explained in the previous section. The proposed system did exceptionally well in the bigger dataset, Ferret-Fc, where the improvement over other systems was at least 11% for the average and over 6% for the maximum recognition rate. The proposed system decreased the storage size on average by 94% from the original cropped image for all of the datasets. Processing time was reduced significantly because the size decreased on average by 94%.

6.4 Conclusions

The experiments used a face identification system using DWT and DCT in three different ways by extracting the features of an image. The system employed a new way of classification by normalizing the $L1$ -Norm distances to determine a person's identity. The DWT coefficients were either truncated, the maximum coefficients were selected, or an iterative method was used to weigh

the feature coefficients to get the most dominant features. The experiments concluded that the proposed system outperformed literature and two independent systems in the Feret-fc and the ORL database for both the average and the maximum recognition rate. In the YALE database, the proposed system was close to the average (and maximum values) and achieved better results only at the maximum recognition rate (six training images). Overall, the proposed system performed very well in the bigger database compared to the systems tested. Future work will consider larger databases and recognition response time.

CHAPTER 7: AN OVERVIEW OF RECENT CONVOLUTIONAL NEURAL NETWORK ALGORITHMS FOR IMAGE RECOGNITION

7.1 Introduction

The safety argument is perhaps the most widely cited in favor of the rapid development and widespread adoption of automated vehicles (AVs)[17]. For autonomous vehicle safety, scene analysis is a fundamental piece. For example, data retrieved from traffic signs, toll information, and vehicle plate numbers are useful in evaluating the surrounding area. Vision-based methods are the basis for a given scene examination and are a vital area of research today. In the context of machine vision, image recognition is the capacity of an algorithm to recognize objects, places, or persons. Object detection is a procedure for identifying a particular object, such as a traffic sign, in a digital image or video. Object recognition algorithms depend on learning or pattern recognition algorithms using feature-based techniques¹.

In the field of image recognition and target detection, convolutional neural networks perform well [166]. They are state of the art. Convolutional neural networks (CNN) can learn hierarchical features, including high-level features, and are a key mechanism for feature extraction [167]. Extracting an image's salient features is the key to successful object recognition, and convolutional neural networks provide a powerful data-driven approach to achieve that goal. The following is a description of the most popular convolutional neural networks. AlexNet is a vast, deep convolutional neural network that won the ImageNet Large Scale Visual Recognition Competition (ILSVRC) in 2010 with a top-5 error of 15.3%, more than 10.8 percentage points better than the second-best. VGGNet is a very deep convolutional network for large-scale recognition. It is known for

¹In this chapter, we partially use the material published in IEEE International Midwest Symposium on Circuits and Systems, 2018 [6].

its simplicity and 3×3 convolutional layers[68]. The most popular branches are the VGG-16 and VGG-19, named for the number of weighted convolution layers. GoogleNet has a drastic change in approach compared to VGGNet and AlexNet. It uses a combination of inception modules, which includes pooling, convolutions with different scales, and concatenation operations. It also uses feature convolutions of 1×1 dimensions that can be considered to be feature selectors[108]. Inception-v3 is the third iteration of the GoogleNet (Inception-v1), which uses the factorization of convolutions and improved normalization. ResNet is a deep residual learning network that learns from residuals as well as learning features. Instead of using several layers stacked one after another, ResNet has an identity shortcut connection that skips one or more layers[109]. ResNet has two popular branches: ResNet-50 (50 layers residual) and ResNet-101 (101 layers residual).

When determining the best image recognition accuracy, considering several aspects of the image is imperative. Depending on the angle the image is taken, images can be focused, unfocused, be bright, or shady due to light and can be tilted. Each of the convolutional neural networks mentioned above was tested using eight different types of pictures with different tilt, focus, and shade.

This paper presents an Object Recognition (OR) system based on the Convolutional Neural Networks mentioned above and performs a comparative evaluation of three different environments using traffic signs as the dataset.

7.2 Proposed System

The experiment used cropped images ($\sim 60 \times \sim 80$ pixels) from the BelgiumTSC for classification dataset as a source for street sign images. As Mathias [9] states, the choice of the datasets is motivated by a large number of annotations, diversity of the content and classes, the availability of a split for benchmarking traffic sign detection, and traffic sign classification separately. All the

data was collected by driving a van with eight cameras on its roof through the streets. About every meter, each of the cameras simultaneously takes a 1628×1236 image. The average speed of the van is 35 km/h. Only traffic signs captured at a distance of less than 50 meters are considered [33].

The training and the testing dataset consists of a total of 62 categories, that were combined. A program was written to gather information about each of the images. The info was tilt, focus, and shading. The image tilt (the angle between 0 and π) was determined by passing it through the Hilbert Transform. Filtering each image using the absolute central moment algorithm written by [168] determines the focus of each image. The image's shade was formulated by the sum of all the pixel colors for the red, green, and blue layers. Sorting of each image in each category by the tilt, focus, and shade created eight possible combinations. The combination was labeled 000, 001, 010, 011, 100, 101, 110, and 111. Where 000 represented images with less than average focus, above-average tilt, and less than average shade. Categories consist of fifteen images fitting the criteria. If there were not enough images to fit in one category, then that category was removed from the dataset.

The experiment used MATLAB R2018a as the development environment and evaluated a convolutional neural network image recognition system that consisted of a feature extraction method and a classification system. The feature extraction model included using several functions found in the Computer Vision Toolbox. The functions used were *imageDatastore*, which creates a structure that holds all the files located in folders and subfolders. The dataset was separated into a training and testing set of images by the function *splitEachLabel*. The function *removeLayers* and *fitcecoc* were also used as well as several others in the Computer Vision Toolbox. The feature extraction methods compared in this experiment were the AlexNet, VGG-16, VGG-19, GoogleNet, Inception-v3 ResNet-50, and ResNet-101.

The least absolute deviation (L1-norm) measurement system was used to measure the distance

between the test images and the trained models. Training utilized ten images, and testing used five images for each category of 15 images. Tests against the eight combinations of the dataset for each of the seven convolutional neural networks mentioned above measured the average recognition rate and recognition time.

7.3 Experimental Results

The number of categories for every eight possible combinations as explained earlier are as follows Dataset111 (25 folders), Dataset110 (15 folders), Dataset101 (18 folders), Dataset100 (25 folders), Dataset011 (7 folders), Dataset010 (1 folders), Dataset001 (22 folders), Dataset000 (28 folders). Table 1 through Table 3 has a summarization of the results of the experiments for recognition rate of the different feature extraction methods.

Three different systems tested each of the datasets and convolutional neural networks. The first system named “System 1” was tested on a network with a mini-batch size of 10, a maximum epoch of 6 (which is one forward pass and one backward pass of all the training examples), the initial learning rate of 0.0001 and validation frequency of 3. The parameters for “System 2” was a network with a mini-batch size of 32, a maximum epoch of 4 (which is one forward pass and one backward pass of all the training examples), the initial learning rate of 0.0001 and validation frequency of 3.

Table 1 shows the feature extraction methods across the eight combinations of categories on “System 2”. The elapsed time for each of the feature extractors (for each dataset) was determined. The time is then divided by the number of categories in the dataset. It is important to note that each of the networks has already been pre-trained, so the time it takes to recognize the test image is on the last three layers of training. Two algorithms tied for slowest algorithms; the AlexNet on dataset

111 and ResNet-101 on dataset 000. It is evident from the table that dataset 010 and 011 took the shortest amount of time because there were only one and seven categories, respectively. The next fastest algorithm across all CNN was the third smallest category at 15 for dataset 110. The pattern follows in the same manner as it took less time to do smaller categories across all CNN. When comparing the eight combinations of datasets for the three criteria: tilt, focus, and shade, a pattern arises. Images that were above average for all three rules or below average for all three rules showed the worst recognition time than all other combinations. It is important to note that the third slowest was the criteria where the image is above average focused but below average in shading and tilt.

Table 7.1: Recognition Time per Folder in seconds for System 2

CNN/ DataSet	000	001	010	011	100	101	110	111
AlexNet	66.1	59.8	33.0	41.1	62.9	54.2	54.7	67.5
VGG -16	65.6	59.1	32.0	40.4	62.5	53.4	50.5	65.3
VGG -19	66.6	60.0	32.0	40.4	63.0	52.8	50.1	62.7
Google- Net	66.0	59.0	33.0	40.1	62.4	53.2	50.5	62.9
Inception -v3	66.2	59.0	34.0	41.1	62.2	52.4	49.5	62.1
ResNet -50	65.2	58.8	33.0	39.9	62.2	52.8	49.9	62.6
ResNet -101	67.5	60.1	33.0	40.6	63.5	53.8	51.1	64.1

Table 7.2: Recognition Accuracy for System 1

CNN/ DataSet	000	001	010	011	100	101	110	111
AlexNet	93.8	92.0	100	100	92.0	94.4	100	99.0
VGG -16	95.5	97.7	100	96.4	95.0	94.4	96.7	99.0
VGG -19	94.6	93.2	100	100	92.0	93.1	96.7	100
Google- Net	92.0	89.8	100	100	86.0	88.9	90.0	97.0
Inception -v3	91.1	94.3	100	100	91.0	93.0	95.0	99.0
ResNet -50	93.8	92.0	100	96.4	88.0	90.3	95.0	100
ResNet -101	95.5	89.8	100	100	94.0	88.9	98.3	100

Table 7.3: Recognition Accuracy for System 2

CNN/ DataSet	000	001	010	011	100	101	110	111
AlexNet	35.7	22.7	100	96.4	50.0	34.7	53.3	75.0
VGG -16	29.5	14.8	100	92.9	49.0	29.2	63.3	77.0
VGG -19	38.4	25.0	100	92.9	50.0	37.5	66.7	80.0
Google- Net	41.7	19.3	100	92.9	45.0	34.7	50.0	80.0
Inception -v3	44.6	10.2	100	92.9	47.0	26.4	50.0	81.0
ResNet -50	41.1	21.6	100	100	57.0	33.3	60.0	79.0
ResNet -101	37.5	29.5	100	100	39.0	27.8	51.7	82.0

Table 2 shows the results for recognition rate of “System 1”. Overall, the average was 95.3, across all datasets it was above 91% and above 92% across all CNN. VGG-16 has the highest average recognition rate of 97.0% across all datasets, and the dataset labeled dataset 111 has the highest recognition rate across all CNN. Please note that dataset 010 is not used as a comparison because it has only one category. It is important to note that even though Dataset011 only has seven categories, it has the worst recognition rate of dataset 111 which has 25 categories.

Table 3 shows the results for the recognition rate of “System 2”. Overall, the average was 58.7,

across all datasets was above 20% and 56% across all CNN. Excluding dataset 010 and dataset 011, the recognition rate for each dataset dropped by approximately 20% compared to that of “System 1” and each of the CNN recognition rate dropped by approximately 30%. The best recognition rate for CNN was ResNet-50 at 61.5%.

7.4 Conclusions

The experiment used CNN and the least absolute deviation to recognize an image through extracting the features using convolutional neural networks: AlexNet, VGG-16, VGG-19, GoogleNet, Inception-v3, ResNet-50, and ResNet-101. The experiments concluded that in a system with an epoch of 4, ResNet-50 had the best accuracy at 61.5%. For an epoch of 6, the best recognition rate was VGG-16 at 97.0% across all datasets. The dataset that had the highest recognition rate overall was the one with above-average focus, tilt, and shade for both tests. Regarding time, training only occurred in the last three layers; therefore was no significant difference between each of the CNN. Regarding datasets, dataset111 had the best recognition time.

CHAPTER 8: TRAFFIC SIGN RECOGNITION BASED ON MULTILAYER PERCEPTRON USING DWT AND DCT

8.1 Introduction

Traffic signs provide information about traffic rules, road conditions, and route directions while assisting drivers for better and safer driving [16]. Accurate and fast recognition of traffic signs is the core of the design of autonomous vehicles.

Traffic signs can be vast and heterogeneous depending on the country, and the datasets containing the various images can be relatively large. At present, several methods have achieved high accuracy without the consideration of time, which is a crucial factor in the real-world applications of traffic sign recognition and can not be ignored [169]. When the essential features of the images are extracted and the original dimensions of the images made smaller, the classification time is shortened, which leads to shorter recognition processing time. This paper highlights a reduction in processing time while maintaining a high level of recognition rate. The focus is on enhancing the critical features of the original image while decreasing its overall size. The principal methodologies include image normalization, feature extraction, and image size reduction. Three different traffic sign datasets were used to compare the system's effectiveness: Belgium Traffic Signs (BTS), Traffic Sign Recognition Dataset (TSRD), and the German Traffic Sign Recognition Benchmark (GTSRB).

Feature extraction and classifier design are two main processing blocks in all pattern recognition and computer vision systems [170]. That is, extracting common feature traits is the key to image recognition. Two feature extractors prevalent in the literature are Discrete Wavelet Transform (DWT), and Discrete Cosine Transform (DCT). DWT is an implementation of the wavelet trans-

form that decomposes an image into a mutually orthogonal set of wavelets [161]. DCT represents an image as a sum of sinusoids of varying magnitudes and frequencies and is often used for energy compaction. It is the feature extraction and compression of DCT that will be useful in the proposed system. Before the feature extraction and image resizing, illumination normalization was performed on the image through anisotropic smoothing. Smoothing reduces image noise without removing significant parts of the image content such as edges, lines, and other details that are important for the interpretation of the image [91].

Neural networks are the leading classifiers in image recognition showing excellent results [171], [172]. Multilayer perceptron (MLPNN) is one of the most straightforward classes of feedforward artificial neural network and is quite useful in recognizing images in large datasets. MLPNN consists of nodes with various weights. The smaller sized input images from the feature extractors mentioned above will both decrease the amount of computation and the number of nodes required in the MLPNN, leading to quicker processing time and faster classification.

In short, the proposed system applied normalization, two algorithms (DWT and DCT), that extracted features from the images as well as compressed the image and a classifier, MLPNN.

The paper is organized as follows: in Section II, there is a detailed explanation of the preprocessing techniques and the classifier. In Section III, the proposed system is described. Section IV comprises of the experimental setup and results, and Section V presents the conclusion.

8.2 Proposed System

The proposed system inputted the images which were reduced in size by the preprocessing system (grayscale and anisotropic smoothing). Since feature extraction is vital to the recognition and pattern matching process, it was the next step to the system. Two favored transforms for extracting

features are DWT and DCT. DWT and DCT are useful in image compression and have shown promise in image recognition. The transforms are both Fourier related transforms where DWT expresses an image regarding a sum of wavelets, whereas DCT expresses an image in terms of a sum of sinusoids with different frequencies and amplitudes.

The two-dimensional DCT input, A , is transformed into the DCT transform, B , which is computed as:

$$B = D * A * D^T \quad (8.1)$$

The coefficients of B are sorted in terms of their energy, where 90% of the maximums are kept while the rest are set to 0. The spatial representation of the new matrix, C , is recovered through the real orthonormal matrix D using equation (8.2).

$$C = D^T * B * D \quad (8.2)$$

The new matrix, C , was inputted into the DWT. The advantages of wavelet transform are that it can capture both frequency and location information for each pixel in the image. The simplest Daubechies wavelets are the Haar wavelets, and they are the most commonly utilized. The Haar transform is used both as an image compression tool and a feature extractor. For feature extraction, the formation of the two-dimensional Haar transform is accomplished by placing the image, C , from the DCT, transform between the Haar matrix, H , and its transpose (H^T) as shown in (8.3)

$$E = H * C * H^T \quad (8.3)$$

90% of the features of E are kept, and the rest were set to 0. The spatial representation of the new matrix is recovered through the real orthonormal Haar matrix H .

CNNs achieve state-of-the-art accuracy on a variety of computer vision tasks, including classification, object localization, detection, recognition, and scene labeling. The proposed system used CNN only after processing the image and taking the extracted features of A and B (as previously described).

The standard CNN framework consists of several convolutional and sub-sampling layers, followed by a fully connected, traditional, multilayer perceptron. CNN algorithms can learn the basis vectors of images and extract useful higher-level features through a hierarchical process. Accordingly, feature maps are obtained by convolution from the learned basis vectors of input images [105].

The proposed system is shown in Figure 8.1. The dataset is comprised of images that are sorted into categories such as stop signs, speed signs, and pedestrian signs. The system inputs 32×32 pixel images from each category and divides them so that 80% is used for training, and 20% is used for testing. The images are grayscaled, filtered using anisotropic diffusion, features are extracted and compressed by DWT/DCT. The compressed images are inputted into the MLPNN, and adjustments are made to the weights and biases of the nodes through stochastic gradient descent backpropagation. The testing images are preprocessed and classified using the predetermined weights and biases of the MLPNN.

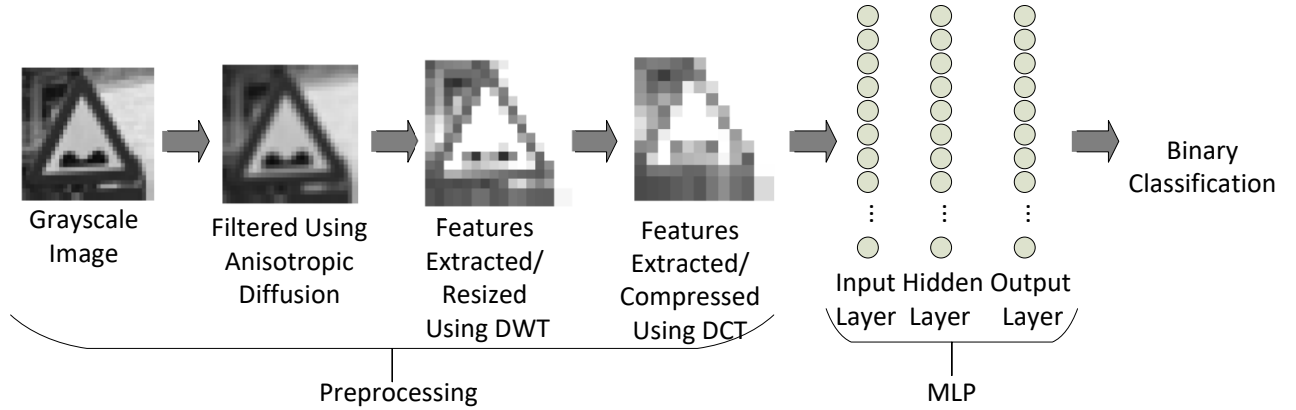


Figure 8.1: Proposed Traffic Sign Recognition System

8.3 Experimental Results

Three different datasets were used to train the system: BTS, TSRD, and GTSRB. BTS is a large dataset with 10000+ traffic sign annotations with thousands of physically distinct traffic signs. The TSRD dataset is a Chinese traffic sign database that includes 6164 traffic sign images containing 58 sign categories. The GTSRB is a large dataset with more than 40 classes and over 50,000 images in total.

For each of the datasets, the images were trained and tested through the MLPNN classifier. The accuracy rate and the processing time was measured for each of the systems and datasets. Systems 2 through 6 are compared to System 1 to show the improvement in speed (due to the decrease in the size of the input array). Accuracy rates, the ratio of processing time and image size are shown in Table 8.1 for the BTS [33], TSRD [173], and GTSRB [155] datasets.

The experiment used MATLAB 2018b as the development environment and evaluated a traffic sign

Table 8.1: Recognition Rate and Comparative Processing Time for BTS, TSRD and GTSRB Datasets

System/Dataset	Accuracy			Relative Processing Time Reduction			Image Size
	BTS	TSRD	GTSRB	BTS	TSRD	GTSRB	All Datasets
System 1	95.7%	94.6%	94.8%	0.00	0.00	0.00	32 x 32
System 2	95.5%	94.3%	94.8%	0.76	0.74	0.76	16 x 16
System 3	94.7%	94.5%	95.0%	0.73	0.72	0.69	16 x 16
System 4	96.0%	94.8%	95.7%	0.71	0.74	0.78	16 x 16
System 5	95.7%	94.9%	95.3	0.76	0.73	0.74	16 x 16
System 6	94.2%	93.8%	94.2	0.85	0.83	0.82	12 x 12

recognition system that consisted of a different combination of the preprocessing step. There are 6 combinations of feature extraction methods compared in this experiment, which is labeled Systems 1 through 6.

This paper looked at six different variations of preprocessing using normalization and DWT/DCT and compared it to System 1 (which has no preprocessing or feature extraction) for base comparison. Each system is labeled in Table 6.1. The first variation is called System 1 and only utilizes image grayscaling with MLPNN as the classifier. The output image had a size of 32×32 . System 2 used normalization and DCT for resizing. The image was converted to the frequency domain using DCT, and coefficients that comprised of 99.99% of the energy were kept and then converted back to the spatial domain. The image was then downsized to 16×16 pixels. System 3 calculated the features LL band of the DWT and consequently decreased the size to 16×16 pixels. Systems 4 and 5 are the same as systems 2 and 3 with anisotropic diffusion before the DCT or DWT step. System 6 used anisotropic diffusion, with cascaded DWT then DCT. The difference is that for DCT, only 99.9% of the energy of the image was kept, and the final size of the image was downsized to 12×12 .

Results for all systems are summarized in Table 6.1. Due to the random nature of the starting

weights, the recognition rate was averaged over five iterations. For systems 2 through 5, there was a relative processing time reduction by 75% and a reduction of 86% for System 6 compared to System 1. Reduction in processing time is due to a decrease of image size from 32×32 to 16×16 or 12×12 (which is 25% or 14 % the size of the original images). The equation used for relative processing time reduction is shown in equation 8.4.

$$1 - \frac{\text{Processing Time System (2 to 6)}}{\text{Processing Time System 1}} \quad (8.4)$$

8.3.1 System Performance Evaluation using the BTS, TSRD and GTSRB Dataset

For all the datasets, the time decreased by 75% compared to the System 1 benchmark (which maintained around 96% accuracy). For the BTS and GTSRB dataset, shown in Table 6.1, System 4 slightly outperformed the other five systems. For the TSRD database, System 5 slightly outperformed the other five systems. In System 6, the accuracy was not considerably lower but had a much higher reduction in processing time.

8.3.2 Remarks on the Performance of the Proposed Systems

The outcome of this experiment unmistakably demonstrated that using a preprocessing algorithm that decreases the image size before an MLPNN, maintains accuracy while decreasing processing time by 25%. Preprocessing with normalization and DWT/DCT feature extraction establishes reasonable grounds for traffic sign recognition. Furthermore, decreasing the searching space while preserving the features of the image increases the speed of the classifier. The disparate systems are trained and tested with three different types of traffic sign datasets, which depicted a wide

variety of traffic signs found on the road. It is clear that with practical and robust image processing techniques, an MLPNN offers one of the best classification approaches.

8.4 Conclusions

In this paper, a traffic sign recognition system was proposed based on preprocessing methods, which enhanced and compressed the image entered into the MLPNN classifier. This preprocessing leads to a better representation of the image with a smaller size. The smaller image meant that fewer nodes were required in the MLPNN and ultimately decreased the processing time of the recognition. The preprocessing step included gray scaling, normalization through anisotropic smoothing, feature extraction, and image resizing/compression using DWT and DCT transforms. An MLPNN was used for classifying. The results of the study indicated that the proposed system that used preprocessing maintained accuracy while decreasing processing time. The decrease in processing time was achieved by using processing techniques that maintained the features of the digital image while being a quarter of the size. This paper compared all systems to the benchmark, System 1, and the system used only a three-layer MLPNN (which is comparatively small). Systems 2 through 6 had outstanding strengths in terms of maintaining accuracy while decreasing processing time. This method was trained and tested with a variety of traffic sign images from different parts of the world, from China, Germany, and Belgium, thus making it more robust for practical implementation.

CHAPTER 9: ROBUST SPEAKER RECOGNITION SYSTEM EMPLOYING COVARIANCE MATRIX AND EIGENVOICE

9.1 Introduction

Speaker recognition systems attempt to recognize a speaker based on features extracted from a speech signal. Features like vocal tract system characteristics, pitch, and information patterns in a text convey speaker information [19]. Short utterance speaker recognition and limiting memory computations have been a focus of interest in many research investigations. In sentences, vowels have a distinct perceptual advantage over consonants in determining intelligibility. They are important in speaker recognition because they do not require much information from the user¹.

In 1991 Turk and Pentland developed the Eigenfaces method based on principal component analysis (PCA) for face recognition [85]. For speech, Kuhn et al. first introduced the concept of Eigenvoices that applies PCA to speaker model parameters, “images,” and applying it to the feature parameters. Eigenvoices is a simple approach to extracting information contained in a collection of speaker traits independent of any for-knowledge of features and uses this information to encode and compare individual speakers. In math terms, the principal components of the distribution of speakers that is the eigenvectors of the covariance matrix of the set of speakers, treating a speaker as a vector in a very high dimensional space. The eigenvectors are ordered, each accounting for a different amount of the variation of the speakers [19]. The speaker model used in this paper will be an “image” consisting of a quantized spectral covariance matrix (unrelated to the covariance matrix used in PCA of the Eigenvoice). Eigenvoice is effective for speaker recognition because it can determine the speaker and represent voices, initially in the space of large dimension, in a

¹In this chapter, we partially use the material published in IEEE International Midwest Symposium on Circuits and Systems, 2013 [4].

low-dimensional linear subspace [86].

The system presented in this paper is text-independent and is divided into four stages, quantized spectral covariance matrix, Eigenvoice, creation of a registered user database of speaker features, and identification using the Mahalanobis distance as shown in Fig. 1. The combination of the spectral covariance matrix and Eigenvoice is the feature extraction portion of the system. However, speech is not inherently 2D, and when simply converted to 2D, the PCA section of the eigenvoice method does not produce good results because of its sensitivity to scaling. In this paper, we propose the conversion of a framed speech signal consisting of only six vowels into a 2D quantized spectral matrix where the features of the speech are extracted using the Eigenvoice mentioned above. The extracted features from the test speech signal are then compared to the registered user database using the Mahalanobis distance, and the closest speaker is identified.

9.2 Proposed System

The steps followed to find the Mel Spectral coefficients are: First, compute the Fourier transform of each window frame of a signal. Second, map the powers of the spectrum each frame onto the Mel scale, using overlapping triangular windows. Third, calculate the logs of the powers at each of the Mel frequencies. Fourth, compute the discrete cosine transform (DCT) of the list of Mel log powers. Lastly, the MFCCs are the amplitudes of the resulting spectrum. The block diagram of the MFCC method is shown in Fig. 9.1.

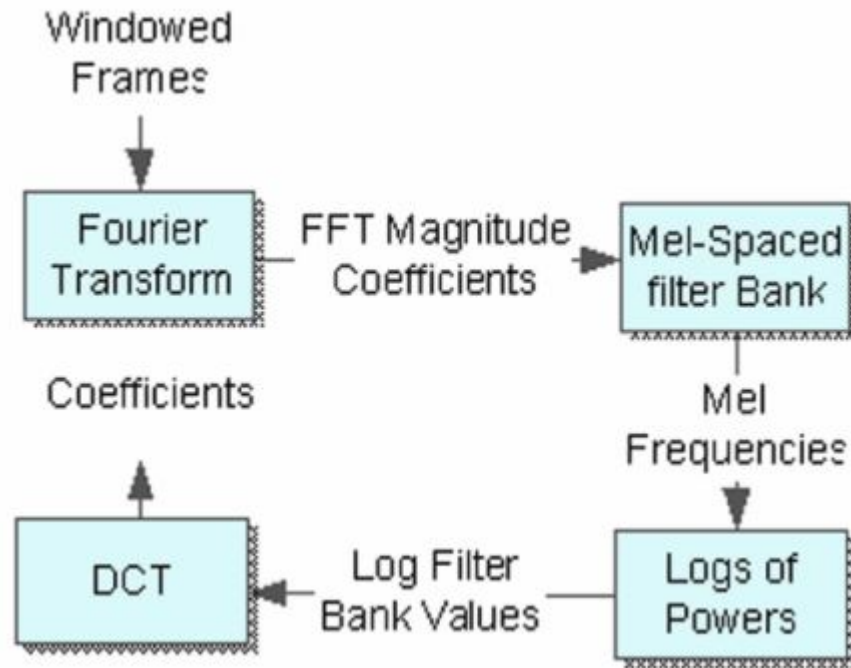


Figure 9.1: Block Diagram of MFCC Feature Extraction Model

The data used in this research is divided into 5 data sets of 24 speakers taken from the TIMIT database. The first data set contains 24 speakers, two males, and one female from 8 dialect regions. The other 4 data sets consist of 24 speakers, of a random number of males and females, of any dialect. Since the first dataset has been set up to be a complete training set by the TIMIT database, it was thought to be a good comparison to a random set of speakers. Further, each data set contains six samples of each of the 24 speakers, and each sample comprises of 6 concatenated vowels.

Three samples of the data set will characterize one speaker in the training phase. The testing data set is generated from the remaining three samples of the concatenated vowels. First, the training set is processed by the quantized spectral covariance matrix algorithm to create a 2D matrix that contains the frequencies with the highest energy from the speech signal. The eigenvoice

coefficients are computed from the combination of all the training speaker data sets. Then the registered user database is created by taking each of the quantized matrices and projecting them on the top 20 eigenvectors to create the basis features of each speaker. The testing phase repeats this process with test data, and the features are extracted by projecting the quantized frequency matrix onto the top 20 eigenvectors. The comparison of the distance between the three samples of the testing features with the user database identifies the speaker. The best out of three correlations is then considered a match and, therefore, the identity of the speaker. See Fig. 9.2 for a flowchart of this process.

9.2.1 Spectral Quantization

Six vowels were chosen to represent each speaker. The vowels are found in the phrase, “She had your dark suit in greasy wash water all year.” The vowels are represented in the database as “iy,” “ae,” “aa,” “ux,” “ih,” and “ao.” These vowels were chosen because they have many samples in the database and have distinct formants compared to each other. These vowels are sampled at 16 kHz and have an average length of 108ms. The training and testing data set consists of these six concatenated vowels at an average length of 668 ms and of length L .

The steps of the spectral quantization are as follows:

1. *Frame and Window the speech signal.* The signal $x_q(n)$ is framed using $N = 512$ point (32 ms) with 50% overlap creating a 2D matrix. Each segment is windowed using the Hamming window so that the segment can be assumed to be stationary in a short time. The Hamming window also reduces the largest side lobe by more than 40 dB down, and therefore it is often a good choice for “1% accurate systems” such as an 8-bit audio signal processing system [174]. Using an overlap of 50% is shown in [175] to have the best accuracy for speaker

recognition, and 512 points give equal accuracy as any smaller point value and decrease the size of the any smaller point value and decrease the size of the matrix. The size of the matrix is $M \times N$ where $M = (Linc)/inc$ where inc is the 50% overlap (256 points) and N is the 512 points. One training dataset will have 72 framed matrices (24 speakers times 3 samples).

2. *Compute the 2D Fast Fourier transform (2DFFT) $X(m, n)$ of the windowed signal.* The Fast Fourier Transform (FFT) is a DFT algorithm which reduces the number of computations. The 2D FFT is the DFT of the rows and columns of the matrix and gives a 2D spectral representation of the speech signal. The 2D FFT used in this paper is $X_q(m, n)$ where X is the data set, $m = 1, \dots, M$; $n = 1, \dots, N$ and $q = 0, \dots, K$ (number of speakers).

Next the magnitude spectrum (9.1) is calculated from the 512 point FFT, with the duplicate portion of the spectrum eliminated due to symmetry; the signal is then divided by the square root of $M \star N$ and the absolute value of the matrix is then calculated. The final signal is of size $m \times p$ where $m = 1, \dots, M$ and p is $1, \dots, N/2$.

$$X_q(m, p) = |X_q(m, n/2)/sqrt(M \star N)| \quad (9.1)$$

3. Covariance matrix to get $R_q(p, p)$. In this case, the matrix X_q is multiplied by its transpose to create a dispersion matrix about the 0 centroid. This matrix is a measure of the spread of the data around the 0 value. The square root of this matrix was taken to find a standard representation of the signal. Note that $R_q(p, p)$ is a symmetric matrix (9.2) where $R_{ij} = R_{ji}$ so the R_{ij} was made to equal 0.

$$R_q(p, p) = \sqrt{X_q(m, p)^T \star X_q(m, p)} \quad (9.2)$$

For each of the five datasets mentioned above, the algorithm was tested in 3 different ways. For the first test, the lower diagonal is removed, and the signal is quantized, as described in

section 4 below. In the second test, the lower diagonal is filled with the kurtosis matrix, where the kurtosis matrix is created by multiplying the covariance matrix (9.2) with its transpose and square rooting the matrix. The matrix was then reshaped to be of size $2 \star N \times N/8$. It was observed that the resulting matrix had a repetitive pattern of relative frequencies, so the third test consisted of a truncated matrix of the second test with a matrix of size $N/2 \times N/8$. Because the high frequencies did not play a role in the identification of the speaker, they were removed for all tests; see eqRefeq:Rq2.

$$R_q(p, r) = R_q(p, p/2) \quad (9.3)$$

The p of the matrix above will be of different size for different tests. In test 1 the $p = 1, \dots, N/4$ and $r = 1, \dots, N/2$. In test 2 the $p = 1, \dots, N$ and $r = 1, \dots, N/8$. In test 3 the $p = 1, \dots, N/4$ and $r = 1, \dots, N/8$.

4. *Saturate all values of $S(n)$ above a threshold $(1 - 2\%)$ and normalize to 1.* It is important to note that PCA is very sensitive to the scaling of the data, so eliminating variations improves performance. The scaled resolution, as seen in (9.4), allows feature identification without losing accuracy and allowing only identifiable features to be kept.

$$\begin{aligned} threshold &= \max(R_q(p, p)) * threshold \\ if(R_q(p, p) > threshold) \\ R_q(p, p) &= threshold \end{aligned} \quad (9.4)$$

The threshold is determined through an iterative empirical process where one sample of the three training samples is tested against the other two. The threshold with the best recognition rate will be used for the testing phase of the algorithm.

5. Quantize $R_q(n)$ to 2 bits by using 4 values 00, 01, 10, 11. The saturated frequency spectrum can be represented as a two-dimensional image with four grayscale colors without degradation to the speaker recognition success rate. A numerical conversion was found to give the same results as a conversion from a 2D matrix to a jpeg image. The conversion scales the image to a certain number of grayscale colors. Experiments revealed that the resolution could be scaled down to only four colors (white, light gray, dark gray, and black) before the ability to identify unique features is degraded and thus negatively impact the speaker recognition success rate. The matrix was normalized, scaled by 3 and then integer quantized to give values of 0, 1, 2, or 3. Eq.(9.5) shows how the scaled matrix is quantized these values. Note that R is a matrix of the same size, as mentioned above.

$$R_q(p, r) = \begin{cases} 0 & \text{if } R_q(p, r) < 0.25 \\ 1 & \text{if } 0.25 \leq R_q(p, r) < 0.5 \\ 2 & \text{if } 0.5 \leq R_q(p, r) < 0.75 \\ 3 & \text{if } 0.75 \leq R_q(p, r) < 1 \end{cases} \quad (9.5)$$

6. *Assemble the training matrix (Eigenvoice coefficients).* For each speaker, the matrix $R_q(p, r)$ in Eq. (9.6) was created and converted to a vector of size $p * r \times 1$. Each of the vectors were assembled together to create the matrix $A(Q, R)$ where Q is $p * r$ and R is the number of

speakers (K) \times 3 samples.

$$\left\{ \begin{array}{l} R_q(p, r) \rightarrow R_q(p * r, 1) \\ A(Q, 1) = R_1(p * r, 1) \\ A(Q, 2) = R_2(p * r, 1) \\ \vdots \\ A(Q, K * 3) = R_{K*3}(p * r, 1) \\ A(Q, R) = [A(Q, 1) \quad A(Q, 2) \quad \dots \quad A(Q, K * 3)] \end{array} \right. \quad \begin{array}{l} \text{for each training R} \\ \\ \\ \\ \\ \end{array} \quad (9.6)$$

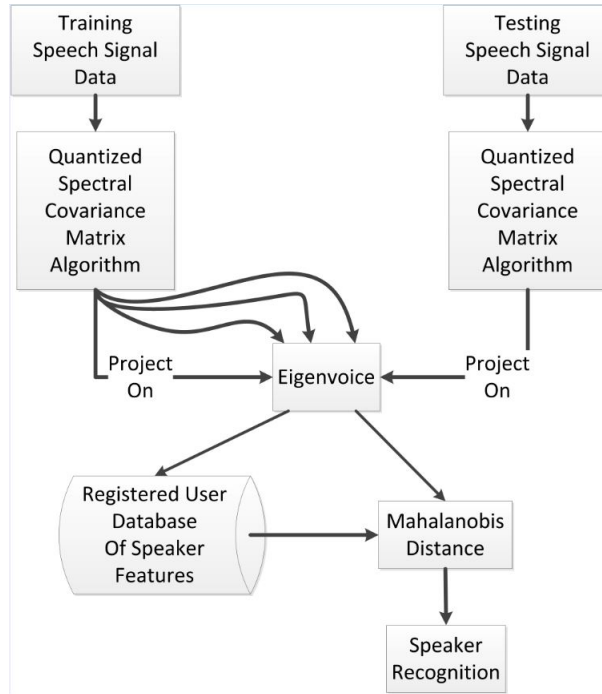


Figure 9.2: Speaker Recognition System

9.2.2 Eigenvoice

PCA finds the vectors that best account for the distribution of speech signals within the entire speech space. In 2DPCA, the input data are represented as matrices rather than vectors. Similar to PCA, it inherits the capability to reduce the dimension of the input data and is therefore widely used in data analysis. Finding the eigenvectors in this way is known as finding the Eigenvoice coefficients of the training set. The calculation method is as follows:

1. *PCA Covariance matrix (S) and eigenvectors:* The covariance matrix S , of $A(n)$ is calculated. Note that this is a different covariance matrix than the one in Eq. eqRefeq:Rq2. This covariance matrix AA^T would be large and of size $p * r \times p * r$ so the linear combinations of the eigenvectors of $A^T A$ is used instead to reduce the matrix to $K \times K$. This reduction is achieved by multiplying A by the eigenvectors of $A^T A$, which gives the eigenvector to AA^T through linear algebra, and the matrix size is reduced.
2. *Eigenvoice Coefficients:* These eigenvectors are in the direction of the largest variance of the training vectors called eigenvoice coefficients. These eigenvoice coefficients can be viewed as features, and when each speaker's voice is projected onto the eigenvoice coefficients, the result describes the importance of each of the features of the voice.

9.2.3 Registered User Database of Speaker Features

To create the database, each of the speakers' principal components (features) is stored in a matrix to be compared to the test speaker's principal features.

9.2.4 Classification

In speaker recognition, the most popular classification method is the Euclidean distance. However, in this paper, the classification method used to compare the training data set with the testing dataset was the Mahalanobis distance. This distance measure was introduced by P. C. Mahalanobis in 1936 [176] and is based on correlations between data sets where different patterns can be identified and analyzed. It measures the similarity of a training data set to a testing data set and is different from the Euclidean distance in that it takes into account the correlations of the data set and is scale-invariant.

There are three samples for each speaker for training and three different samples for testing. The algorithm used to determine a match between the training set and the testing set was that of best two out of three. Where each sample was compared to the training dataset and if two out of the three samples matched a speaker it was determined that this was the original speaker.

9.3 Experimental Results

In this experiment, the proposed speaker recognition system is evaluated with five datasets of 24 speakers (six vowels and six samples for each speaker). In this new approach, each of the datasets was tested using different variance matrix as described previously: the variance matrix only, using the added kurtosis matrix, and using a truncated kurtosis matrix. The results are shown in Fig. 9.3 and Fig. 9.4. Overall, the first dataset had the best recognition rate and is a good comparison test. For most datasets, the addition of kurtosis gave better results than variance alone. Truncating the combination of variance/kurtosis matrix did not significantly affect the results, but it did decrease the size of the matrix and therefore decrease memory storage space and increase computational speed. At various SNR, in this noisy environment, the high recognition rate was achieved where

the SNR did not degrade performance significantly from 25dB to 0dB.

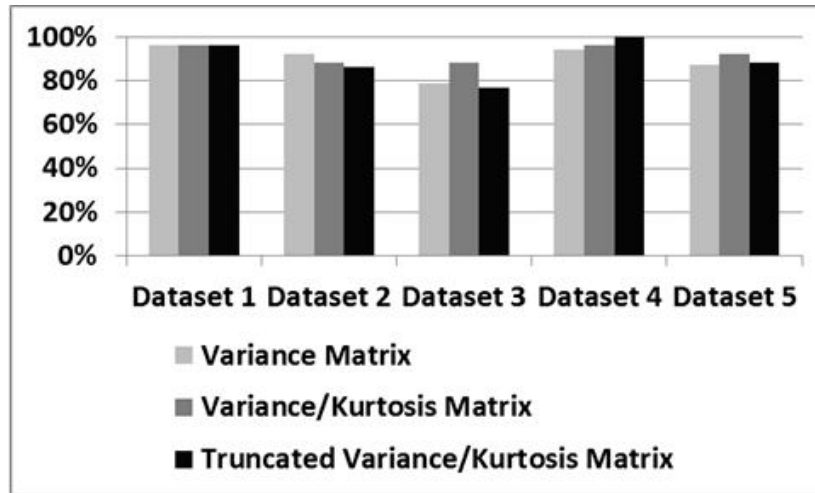


Figure 9.3: Average Recognition Rate of the Five Datasets Using Various Matrices

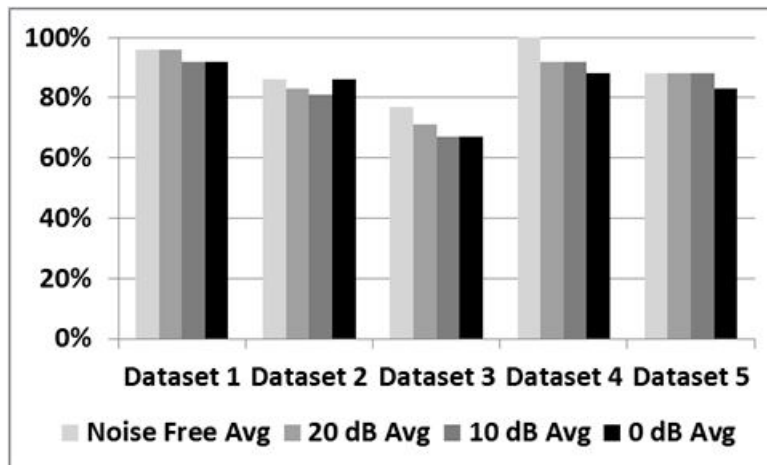


Figure 9.4: Recognition Rate of the Five Datasets under Train Noise for Various SNR

9.4 Conclusions

A novel speaker recognition system using a quantized spectral covariance matrix with Eigenvoice was proposed in this paper and shown to have a very high recognition rate in the presence of noise at varying levels in five datasets of 24 speakers. The novel contribution was to use the covariance matrix and data limitations as a 2D matrix for recognizing a speaker. The Fourier transform, and Eigenvoice is used for feature extraction while the Mahalanobis distance is used for recognizing the speaker. Future work includes using a larger dataset that would determine the viability of this method compared to Mel Frequency Cepstrum Coefficients (MFCC) and various databases like YOHO.

CHAPTER 10: AN OVERVIEW OF RECENT WINDOW BASED FEATURE EXTRACTION ALGORITHMS FOR SPEAKER RECOGNITION

10.1 Introduction

Speaker recognition can be divided into two components: speaker identification and speaker verification. Speaker identification identifies the speaker that has most likely spoken from a group or population based on the features of a speech signal. Speaker verification accepts or rejects the identity of the speaker based on the sample speech signal. Speaker recognition can be further divided into text-dependent and text-independent systems. A text-dependent system requires the speaker to repeat a specific phrase that has been used in training the system, while a text-independent system is based on the shape and size of the vocal tract, dynamics of the articulators, rate of vibration of the vocal folds, accent imposed by the speaker, and speaking rate [177]¹

Extracting a speaker's feature traits is the key to speaker recognition. There are several feature extractors popular in literature, which are briefly described here. Linear Predictive Coefficients (LPC) models the vocal tract by using an all-pole model, Linear Predictive Cepstral Coefficients (LPCC) captures the main vocal tract resonance property in the acoustic spectrum, Mel Cepstral Coefficients (MFCC) obtains a set of filters which is similar to the role of the cochlea. Delta MFCC (Δ MFCC) captures the transitional characteristics of the speech signal. Perceptual Linear Predictive Cepstrum Coefficients (PLPCC) is based on a short-term spectrum of speech. Real Cepstral Coefficients (RCC) characterizes the shape of the vocal tract at the current frame. This

¹In this chapter, we partially use the material published in IEEE International Midwest Symposium on Circuits and Systems, 2018 [3].

paper presents a speaker identification system based on the Vector Quantization (VQ) and performs a comparative evaluation of the feature, as mentioned earlier, extraction methods using a text-dependent system in a noise-free environment.

10.2 Proposed System

The experiment used the Center for Spoken Language Understanding (CLSU) Speaker Recognition Corpus data collection (version 1.1) as a source for the speaker's speech signal. As Cole [178] states, this database is a collection of telephone speech recordings from over 500 participants collected over two years. The protocol includes fixed vocabulary phrases, digit strings, personal utterances, and fluent speech [178]. All the data was collected over the telephone line and were sampled at 8 kHz 8 bit, 16 bit linearly encoded, and stored as μ law files. 64 speakers were chosen from the database where they were recorded saying the following three phrases "Charlie did you think to measure the tree?", "Pasadena, California" and "mango." The phrase was repeated twice by each speaker in the same session, and one occurrence was used for training, and the other occurrence was used for testing. The setup of the experiment is shown in Fig 10.1.

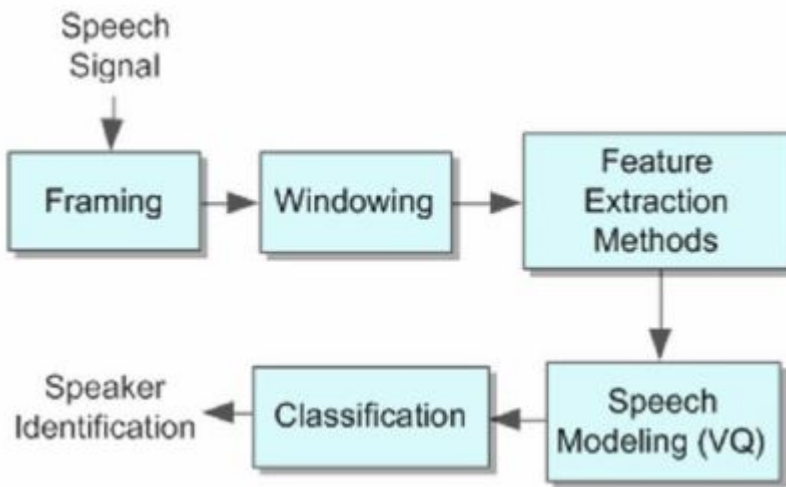


Figure 10.1: Experimental Setup of Speaker Identification System

The experiment used MATLAB 11.0 as the development environment and evaluated a speaker identification system that consisted of a feature extraction method, a model system, and a classification system. Feature extraction methods compared in this experiment were the MFCC, MFCC+ Δ MFCC, RCC, LPC, LPCC, PLPCC, and Rasta-PLPCC.

Vector Quantization was the model system, and the Euclidean measurement system was used to measure the distance between the test utterance and the trained models. The frame length and the overlap used for windowing were 16 ms and 50% shown in Fig. 10.2. The window type was the Hamming window for all methods except for the LPC, where the Hanning window was used. A total of 12 centroids were used, and the number of coefficients used for MFCC was also 12. Each method was tested five times for each of the 64 test speakers, and the average recognition rate and recognition time was measured.

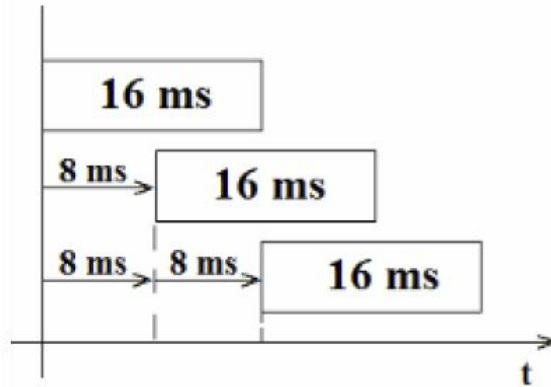


Figure 10.2: Frame Length and Overlap

10.3 Experimental Results

Results of the experiments for recognition rate of the different feature extraction methods are summarized in Table 10.1 and Fig 10.3. By itself, MFCC outperforms all other methods when considering both recognition rate and recognition time. This performance supports the findings of Kumar [177], where MFCC features showed a better identification rate compared to LPC. The addition of the Δ MFCC to MFCC showed little improvement to the recognition rate supporting the findings of Hossan [179], which found an improvement of only 1% over MFCC. LPCC and PLPCC methods were the second-best methods with a recognition rate of 91% and 90%, respectively. LPCC came in second to MFCC, and Δ MFCC is supporting the work of Biswas [180] where LPCC was the feature of the second choice. RCC and LPC methods were the worst methods with a recognition rate of 84% and 81%, respectively. This performance is supported by Islam [112], where RCC and LPCC were shown to have a poor recognition rate.

Table 10.1: Recognition Rate of Feature Extraction Methods

Phrase/Methods	Δ MFCC	MFCC	LPCC	PLPC	RCC	LPC	Rasta-PLPC
Charlie ... ?	98%	98%	99%	92%	89%	88%	27%
Pasadena ...	98%	98%	92%	98%	88%	88%	20%
Mango	87%	87%	81%	81%	75%	68%	23%

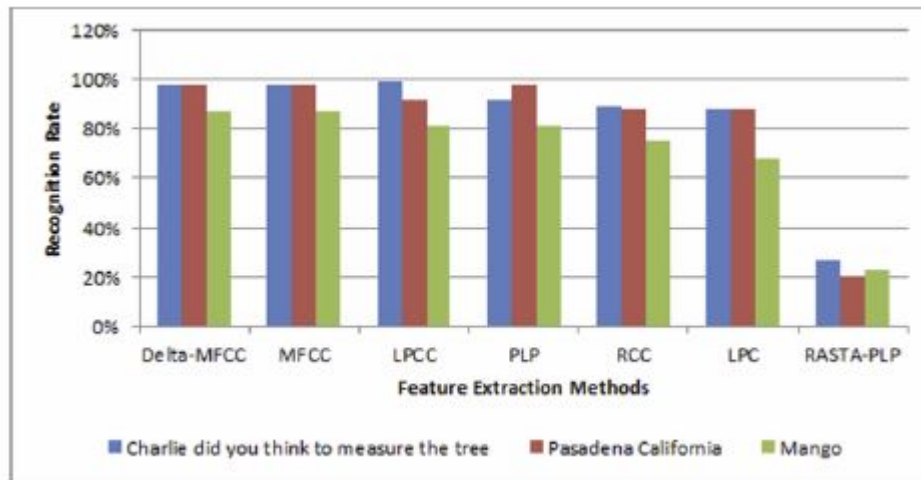


Figure 10.3: Recognition Rate of Feature Extraction Methods

The RASTA-PLPCC method was found to be very ineffective in identifying the speaker. It could be argued that Schurer [181] supports this, where it was shown that RASTA-PLPCC worked best with either Hidden Markov Model (HMM) or MultiLayer Perceptron (MLPNN) models and not with VQ models. When comparing the trend of the recognition rate of the three different phrases used, the first two phrases (Charlie... and Pasadena ...) had very similar recognition rates, but the phrase “Mango” had a noticeably lower recognition rate. This result shows that to identify

a speaker effectively, the length of the speech signal needs to be longer than the length of the word “Mango.” In terms of recognition times, shown in Fig. 10.4, MFCC was faster than all other methods, and RCC was the slowest by 2.5 times that of MFCC.

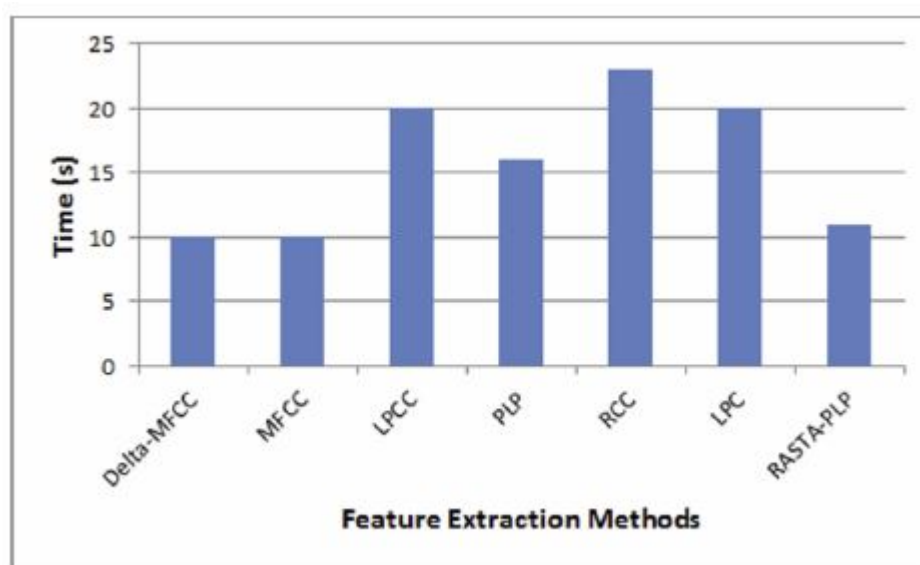


Figure 10.4: Feature Extraction Methods

10.4 Conclusions

The experiment used VQ and Euclidean distance to recognize a speaker’s identity through extracting the features of the speech signal by the following methods: RCC, MFCC, MFCC + Δ MFCC, LPC, LPCC, PLPCC, and RASTA PLPCC. The experiments concluded that, in a noise-free environment MFCC, (at an average of 94%), is the best feature extraction method when used in conjunction with the VQ model. The addition of the delta MFCC showed no significant improvement to the recognition rate. When comparing the three different phrases, the longer two phrases had a very similar recognition rate, but the shorter-phrase at 0.5 seconds had a noticeable lower

recognition rate. This result clearly shows that for an effective speaker identification, the speech sample length needs to be longer than a single two-syllable word length. When comparing recognition time, MFCC was faster than all other methods. Overall, MFCC in a noise-free environment was the best method in terms of recognition rate and recognition rate time. More in-depth results will be presented later with more severe conditions like a noisy environment, a text-independent system, and a fusion of feature extraction methods.

CHAPTER 11: CONCLUSION AND FUTURE WORK

When designing any recognition system, four factors need to be considered: recognition accuracy, computational complexity, computational speed, and storage requirements. Most of the recognition systems found in the literature focus only on attaining higher recognition rates; we focused on all four factors. Those systems assume that the higher computational complexity and larger storage size are issues that can be resolved using other readily available signal processing techniques. Here, we addressed all four aspects.

Pattern recognition systems are composed of three main stages: preprocessing, feature extraction, and classification. Depending on the recognition systems (face, traffic sign, and voice), the proposed system contained improvements to one or all of these three stages. For facial recognition, grayscaling and anisotropic smoothing were considered to improve the preprocessing step. In the feature extraction, 2D-DCT and 2D-DWT were tested adaptively or recursively to improve the system. In the classification section, the MLSNN, MLPNN, and CNN were tested as both a feature extractor and a classifier to get the best system in terms of accuracy, storage space, and speed. In speaker recognition, we focused on feature extraction by testing a covariance matrix mixed with the vector quantization. We also experimentally determined the optimal sample size in the context of several specific datasets. In addition, we studied the popular convolutional neural networks as well as the popular window-based feature extractors. In each of these applications, we demonstrated that the system could maintain high accuracy while keeping the processing speed high, storage requirements low, and processing complexity low.

The ORL, YALE, FERET-fc, FEI, MNIST datasets were used to evaluate the performance of the proposed systems in chapter 4. For the system described in chapter 5, the datasets used to evaluate the performance of those systems were ORL, YALE, and FERET-fc. In chapter 6, the BTS

dataset was used to evaluate the performance of systems. In chapter 7, the datasets BTS, GTSRB, and TSRD were used to evaluate the performance of the system. In chapter 8, the TIMIT and NOIZEOUS datasets were used to evaluate the performance of the system. Finally, in chapter 9, the CLSU corpus data collection was used to test the performance of the system.


The MATLAB environment was the principal software environment used to evaluate the performance of the proposed techniques. The experimental results confirmed the improved performance of the proposed systems. Specifically, we demonstrated improvements in terms of better recognition accuracy, reduced storage requirements, increased processing speed, and reduced computational burden as compared to some recently reported approaches.

11.1 Future Work

Several suggestions for future work are listed in this section.

- Based on the results we obtained for Face Recognition, nonorthogonal mixed transforms using various popular transforms in addition to DWT and DCT will be used.
- To realize a more realistic scenario, the proposed techniques will be tested on larger datasets.
- Although some datasets had different lighting and partial occlusion, we would like to test the techniques found here with various datasets with different lighting, partial occlusion, and in a noisy environment.
- The proposed recognition techniques here have been tested on the face, traffic sign, and handwritten digits, but we would like to expand this technique to medical images such as Electroencephalogram (EEG) signals.

**APPENDIX A: PERMISSION LETTERS TO REPRINT THE ARTICLES
IN THIS DISSERTATION**



An Overview of Recent Convolutional Neural Network Algorithms for Image Recognition

Conference Proceedings: 2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)

Author: Genevieve Saplijaszko

Publisher: IEEE

Date: Aug, 2018

Copyright © 2018, IEEE

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

- 1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
- 2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
- 3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

- 1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
- 2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
- 3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.


If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK
CLOSE

© 2020 Copyright - All Rights Reserved | Copyright Clearance Center, Inc. | [Privacy statement](#) | [Terms and Conditions](#)

Comments? We would like to hear from you. E-mail us at customer-care@copyright.com

Figure A.1: Reprint permission request for the Recent Convolutional Neural Networks Article



Traffic Sign Recognition Based on Multilayer Perceptron Using DWT and DCT

Conference Proceedings: 2019 IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS)

Author: Genevieve Sapliaszko

Publisher: IEEE

Date: Aug. 2019

Copyright © 2019, IEEE

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

- 1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
- 2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
- 3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

- 1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
- 2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
- 3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.


If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK
CLOSE

© 2020 Copyright - All Rights Reserved | Copyright Clearance Center, Inc. | Privacy statement | Terms and Conditions

Comments? We would like to hear from you. E-mail us at customer-care@copyright.com

Figure A.2: Reprint permission request for the Traffic Sign Recognition System Article



Adaptive Feature Extraction Algorithm using Mixed Transforms for Facial Recognition

Conference Proceedings: 2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)

Author: Genevieve Sapliaszko

Publisher: IEEE

Date: Aug. 2018

Copyright © 2018, IEEE

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

- 1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
- 2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
- 3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

- 1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
- 2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
- 3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.


If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK
CLOSE

© 2020 Copyright - All Rights Reserved | Copyright Clearance Center, Inc. | Privacy statement | Terms and Conditions

Comments? We would like to hear from you. E-mail us at customer-care@copyright.com

Figure A.3: Reprint permission request for the Adaptive Feature Extraction Algorithm Article



An overview of recent window based feature extraction algorithms for speaker recognition

Conference Proceedings: 2012 IEEE 55th International Midwest Symposium on Circuits and Systems (MWSCAS)

Author: Genevieve I. Sapliaszko

Publisher: IEEE

Date: Aug, 2012

Copyright © 2012, IEEE

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

- 1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
- 2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
- 3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

- 1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
- 2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
- 3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.


If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK
CLOSE

© 2020 Copyright - All Rights Reserved | Copyright Clearance Center, Inc. | Privacy statement | Terms and Conditions

Comments? We would like to hear from you. E-mail us at customer-care@copyright.com

Figure A.4: Reprint permission request for the Overview of Recent Window Based Systems Article



Robust speaker recognition system employing covariance matrix and Eigenvoice

Conference Proceedings: 2013 IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS)

Author: Genevieve I. Sapliaszko

Publisher: IEEE

Date: Aug, 2013

Copyright © 2013, IEEE

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

- 1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
- 2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
- 3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

- 1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
- 2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
- 3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK
CLOSE

© 2020 Copyright - All Rights Reserved | Copyright Clearance Center, Inc. | Privacy statement | Terms and Conditions

Comments? We would like to hear from you. E-mail us at customer-care@copyright.com

Figure A.5: Reprint permission request for the Robust Speaker Recognition Article

APPENDIX B: SAMPLE SIZE DETERMINATION SOFTWARE CODE

Feature Extraction Matlab code

% AlexNet Feature Extraction on Belgium Database

```
cat = 62;
```

```
imgDim = 32;
```

```
percentTrain = 0.7;
```

```
samples = 5;
```

```
strFolder = 'BelgiumTSC_Mixed_complete10';
```

```
imds = imageDatastore(strFolder, 'IncludeSubfolders', true, ...  
'LabelSource', 'foldernames');
```

```
tic
```

```
imds30 = splitEachLabel(imds, samples, 'randomized');
```

```
[imdsTrain, imdsTest] = splitEachLabel(imds30, percentTrain, ...  
'randomized');
```

```
net = alexnet;
```

```
layersTransfer = net.Layers(1:end-3);
```

```
numClasses = numel(categories(imdsTrain.Labels));
```

```
newLayers = [
```

```

layersTransfer
fullyConnectedLayer(numClasses,'WeightLearnRateFactor',20,
'BiasLearnRateFactor',20)
softmaxLayer
classificationLayer];

pixelRange = [-30 30];
imageAugmenter = imageDataAugmenter( ...
    'RandXReflection',true, ...
    'RandXTranslation',pixelRange, ...
    'RandYTranslation',pixelRange);

inputSize = net.Layers(1).InputSize;
augimdsTrain = augmentedImageDatastore(inputSize(1:2), ...
imdsTrain, 'DataAugmentation',imageAugmenter);
augimdsTest = augmentedImageDatastore(inputSize(1:2), ...
imdsTest);

options = trainingOptions('sgdm', ...
    'MiniBatchSize',10, ...
    'MaxEpochs',6, ...
    'InitialLearnRate',1e-4, ...
    'ValidationData',augimdsTest, ...
    'ValidationFrequency',3, ...
    'ValidationPatience',Inf, ...

```

```

    'Verbose', false, ...
    'Plots', 'none');%, 'ExecutionEnvironment', 'parallel');

netTransfer = trainNetwork(augimdsTrain, newLayers, options);

[YPred, probs] = classify(netTransfer, augimdsTest);
accuracy = mean(YPred == imdsTest.Labels);
disp(['accuracy - ' num2str(accuracy*100)]);
elapsed = toc
str = ['McDowellAccuracyB' '-' num2str(samples)];

%%
numTrainImages = size(imdsTrain.Files, 1);
trainLabels = zeros(cat, numTrainImages);
trainImages = zeros(imgDim, imgDim, numTrainImages);
clear newImg;
for i = 1:numTrainImages %Go through all categories
    clear newImg newImgN img2;
    [img, fileInfo] = readimage(imdsTrain, i);
    img2 = rgb2gray(img);
    newImg = double(img2(:, :, 1))/255.0;
    trainImages(:, :, i) = imresize(newImg, [imgDim, imgDim]);
    row = floor((i-1)/(percentTrain*samples))+1;
    trainLabels(row, i) = 1;
end

```

```

numTestImages = size(imdsTest.Files,1);
testLabels = zeros(cat,numTestImages);
testImages = zeros(imgDim,imgDim,numTestImages);
clear newImg;
for i = 1:numTestImages %Go through all categories
    clear newImg newImgN img2;
    [img,fileinfo] = readimage(imdsTest,i);
    img2 = rgb2gray(img);
    newImg = double(img2(:,:,1))/255.0;
    testImages(:,:,i) = imresize(newImg,[imgDim,imgDim]);
    row = floor((i-1)/((1-percentTrain)*samples))+1;
    testLabels(row,i) = 1;
end

%%
mcovariance = cov(trainImages(:));
mstdDev = std(trainImages(:));
mvvariance = var(trainImages(:));
mkurtosis = kurtosis(trainImages(:));
mskewness = skewness(trainImages(:));
msum = sum(trainImages(:));
mmean = mean(trainImages(:));

save(str, 'accuracy' , 'elapsed' , 'samples' , 'mcovariance' , ...

```

```
'mstdDev', 'mvariance', 'mkurtosis', 'mskewness', 'msum', ...  
'mmean');
```

Sample Size Determination Sample Code

%Calculating the sample size based on database

```
percentTrain = 0.9;  
error_tolerance = 0.05;  
hiddenlayersize1 = 3;  
hiddenlayersize2 = 3;  
numSamples = 10;  
categories = 62;  
imgDim = 32;  
inputlayersize = 13;
```

%set non-random seed

```
rng('default');  
rng(1);
```

%% input data

```
filename = 'BelgiumStat.csv';  
dData = xlsread(filename);
```

%Normalize Data

```
mindData = min(dData) ;
```

```

normA = dData - mindData;
maxdData = max(normA);
Data = normA ./ maxdData;

% create training and testing matrices
[entries , attributes] = size(Data);
numTrain = round(entries*percentTrain);
outputlayersize = attributes - inputlayersize;
trainingdata = Data(1:numTrain,:);
trainingdata_inputs = trainingdata(:,1:inputlayersize);
trainingdata_outputs = trainingdata(:,inputlayersize+1:end);
testingdata = Data(numTrain+1:end,:);
testingdata_inputs= testingdata(:,1:inputlayersize);
testingdata_outputs= testingdata(:,inputlayersize+1:end);

%% Initialize random synapse weights with a mean of 0
syn0 = 2*rand(inputlayersize , hiddenlayersize1) - 1;
syn1 = 2*rand(hiddenlayersize1,hiddenlayersize2) - 1;
syn2 = 2*rand(hiddenlayersize2,outputlayersize) - 1;

%% Feedforward training data
layer0=trainingdata_inputs;
layer1=(1)./(1+exp(-1.*(layer0*syn0)));
layer2=(1)./(1+exp(-1.*(layer1*syn1)));
layer3=(1)./(1+exp(-1.*(layer2*syn2)));

```

```

%check for accuracy

err = immse(layer3, trainingdata_outputs);
accuracy = (1 - err)*100;
fprintf("Untrained: Mean Squared Error with Training Data: ...
%f Accuracy: %f hidden layer size: %f\n", err, accuracy, ...
hiddenlayersize1)

```

```

%% Feedforward testing data

```

```

layer0=testingdata_inputs;

layer1=(1)./(1+exp(-1.*(layer0*syn0)));
layer2=(1)./(1+exp(-1.*(layer1*syn1)));
layer3=(1)./(1+exp(-1.*(layer2*syn2)));

```

```

%Check for accuracy

err = immse(layer3, testingdata_outputs);
accuracy = (1 - err)*100;
fprintf("Untrained: Mean Squared Error with Testing Data: ...
%f Accuracy: %f hidden layer size: %f\n", err, accuracy, ...
hiddenlayersize1)

```

```

%% Best alpha for data = 0.001

```

```

for alpha = 0.001
    fprintf("Training with alpha: %f\n", alpha)

```



```

for iter = 1:1000000
    %Feedforward

    layer0 = trainingdata_inputs;
    layer1=(1)./(1+exp(-1.*(layer0*syn0)));
    layer2=(1)./(1+exp(-1.*(layer1*syn1)));
    layer3=(1)./(1+exp(-1.*(layer2*syn2)));

    %Back-propagation

    l3_error = layer3 - trainingdata_outputs;
    l3_delta = l3_error.*(exp(layer3)./(exp(layer3)+1).^2);
    l2_error = l3_delta*syn2.';
    l2_delta = l2_error.*(exp(layer2)./(exp(layer2)+1).^2);
    l1_error = l2_delta*syn1.';
    l1_delta = l1_error.*(exp(layer1)./(exp(layer1)+1).^2);

    %Adjust values

    errorval = mean(abs(l3_error));
    syn2 = syn2 - alpha.*(layer2.'*l3_delta);
    syn1 = syn1 - alpha.*(layer1.'*l2_delta);
    syn0 = syn0 - alpha.*(layer0.'*l1_delta);

    if errorval < error_tolerance
        fprintf("Stopping at: %f error\n", errorval)
        break

```

```

end

%print out debug data
    if iter==1 || mod(iter,25000) == 0
        accuracy = (1-errorval)*100;
        fprintf("\titer=%f, Error: %f, Accuracy: ...
%f\n", iter, errorval, accuracy)
    end
end

if errorval > error_tolerance
    fprintf("A value below the tolerance not found, ...
please change alpha\n\n")
else
    accuracy = (1-errorval)*100;
    fprintf("A value below the tolerance found: ...
%f Accuracy %f\n\n", errorval, accuracy)
end
end

end

%Feedforward training data
layer0=trainingdata_inputs;
layer1=(1)./(1+exp(-1.*(layer0*syn0)));
layer2=(1)./(1+exp(-1.*(layer1*syn1)));
layer3=(1)./(1+exp(-1.*(layer2*syn2)));

```

```

sampleResultsInput = round(layer3 .* ...
maxdData(inputlayersize+1) + mindData(inputlayersize+1));

%check for accuracy
err = immse(layer3, trainingdata_outputs);
accuracy = (1-err)*100;
fprintf("Trained: Mean Squared Error with Training Data: ...
%f Accuracy: %f\n", err, accuracy);

%feedforward testing data
layer0=testingdata_inputs;
layer1=(1)./(1+exp(-1.*(layer0*syn0)));
layer2=(1)./(1+exp(-1.*(layer1*syn1)));
layer3=(1)./(1+exp(-1.*(layer2*syn2)));

%check for accuracy
err = immse(layer3, testingdata_outputs);
accuracy = (1-err)*100;
fprintf("Trained: Mean Squared Error with Testing Data: ...
%f Accuracy: %f\n", err, accuracy);

%Results - Samples
sampleResultsOutput = round(layer3 .* ...
maxdData(inputlayersize+1) + mindData(inputlayersize+1));

```

LIST OF REFERENCES

- [1] Kenneth O. Stanley. Neuroevolution: A different kind of deep learning, July 2017.
- [2] Genevieve Sapijaszko, Taif Alobaidi, and Wasfy B Mikhael. Traffic sign recognition based on multilayer perceptron using dwt and dct. In *2019 IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 440–443. IEEE, 2019.
- [3] Genevieve I Sapijaszko and Wasfy B Mikhael. An overview of recent window based feature extraction algorithms for speaker recognition. In *2012 IEEE 55th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 880–883. IEEE, 2012.
- [4] Genevieve I Sapijaszko and Wasfy B Mikhael. Robust speaker recognition system employing covariance matrix and eigenvoice. In *2013 IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 1116–1119. IEEE, 2013.
- [5] Genevieve Sapijaszko, Taif Alobaidi, and Wasfy B Mikhael. Adaptive feature extraction algorithm using mixed transforms for facial recognition. In *2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 226–229. IEEE, 2018.
- [6] Genevieve Sapijaszko and Wasfy B Mikhael. An overview of recent convolutional neural network algorithms for image recognition. In *2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 743–746. IEEE, 2018.
- [7] Genevieve Sapijaszko and Wasfy B Mikhael. Designing cnn for various datasets using neuroevolution. *Neurocomputing*, -(-):-, 2020.
- [8] Genevieve Sapijaszko and Wasfy B Mikhael. Wavelet based methods in facial recognition using mixed transform and convolutional neural networks. In *2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages –. IEEE, 2020.

- [9] Genevieve Sapijaszko and Wasfy B Mikhael. Predicting optimal sample size for classification performance using convolutional neural network. In *2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages –. IEEE, 2020.
- [10] Genevieve Sapijaszko and Wasfy B Mikhael. Adaptive mixed transforms and convolutional neural networks for image recognition. In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages –. IEEE, 2021.
- [11] Genevieve Sapijaszko and Wasfy B Mikhael. Adaptive mixed transforms with adaptive neural networks for image recognition. In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages –. IEEE, 2021.
- [12] Wai Kai Chen. *The electrical engineering handbook*. Elsevier, 2004.
- [13] Chris Solomon and Toby Breckon. *Fundamentals of Digital Image Processing: A practical approach with examples in Matlab*. John Wiley & Sons, 2011.
- [14] Damilola Omoyiwola. Machine Learning on Facial Recognition, November 2018.
- [15] Jonnathann Finizola, Jonas Targino, Felipe Teodoro, and Clodoaldo Lima. Comparative study between deep face, autoencoder and traditional machine learning techniques aiming at biometric facial recognition. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 07 2019.
- [16] M Swathi and KV Suresh. Automatic traffic sign detection and recognition: A review. In *2017 International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies (ICAMMAET)*, pages 1–6. IEEE, 2017.
- [17] Daniel J Hicks. The safety of autonomous vehicles: Lessons from philosophy of science. *IEEE Technology and Society Magazine*, 37(1):62–69, 2018.

- [18] J.P. Campbell. Speaker recognition: a tutorial. *Proceedings of the IEEE*, 85(9):1437–1462, September 1997.
- [19] Lawrence R Rabiner, Ronald W Schafer, et al. Introduction to digital speech processing. *Foundations and Trends® in Signal Processing*, 1(1–2):1–194, 2007.
- [20] Sadaoki Furui. *Digital speech processing: synthesis, and recognition*. CRC Press, 2018.
- [21] Cheryl Rosen. Voice Recognition To Ease Travel Bookings: Business Travel News.
- [22] Seyyede Samine Hosseini and Shahriar Mohammadi. Review banking on biometric in the world’s banks and introducing a biometric model for iran’s banking system. *Journal of Basic and Applied Scientific Research*, 2(9):9152–9160, 2012.
- [23] Nadia Khomami. Mohammed Emwazi: who were his victims? *The Guardian*, November 2015.
- [24] Thomas Fang Zheng and Lantian Li. *Robustness-related issues in speaker recognition*. Springer, 2017.
- [25] Miguel De-la Torre, Eric Granger, Paulo VW Radtke, Robert Sabourin, and Dmitry O Gorodnichy. Partially-supervised learning from facial trajectories for face recognition in video surveillance. *Information Fusion*, 24:31–53, 2015.
- [26] Amal Azazi, Syaheerah Lebai Lutfi, Ibrahim Venkat, and Fernando Fernández-Martínez. Towards a robust affect recognition: Automatic facial expression recognition in 3d faces. *Expert Systems with Applications*, 42(6):3056–3066, 2015.
- [27] Ü Çigdem Turhal and Alpaslan Duysak. Cross grouping strategy based 2dpca method for face recognition. *Applied Soft Computing*, 29:270–279, 2015.

- [28] ORL Database. At&t laboratories cambridge database of faces. <http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>, April (1992-1994).
- [29] et al. Georgiades, A. Yale face database. center for computational vision and control at yale university. <http://vision.ucsd.edu/content/yale-face-database>, 1997. Accessed: 2019-09-30.
- [30] P Jonathon Phillips, Harry Wechsler, Jeffery Huang, and Patrick J Rauss. The feret database and evaluation procedure for face-recognition algorithms. *Image and vision computing*, 16(5):295–306, (1998).
- [31] P Jonathon Phillips, Hyeonjoon Moon, Syed Rizvi, Patrick J Rauss, et al. The feret evaluation methodology for face-recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1090–1104, (2000).
- [32] Carlos Eduardo Thomaz and Gilson Antonio Giraldi. A new ranking method for principal components analysis and its application to face image analysis. *Image and Vision Computing*, 28(6):902–913, 2010.
- [33] Markus Mathias, Radu Timofte, Rodrigo Benenson, and Luc Van Gool. Traffic sign recognition—how far are we from the solution? In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–8. IEEE, 2013.
- [34] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [35] Jianfen Ma, Yi Hu, and Philipos C Loizou. Objective measures for predicting speech intelligibility in noisy conditions based on new band-importance functions. *The Journal of the Acoustical Society of America*, 125(5):3387–3405, 2009.

- [36] James P Keener. *Principles of applied mathematics: transformation and approximation*. CRC Press, 2019.
- [37] Hari Kumar Singh, Prashant Kr Maurya, Khushboo Singh, and Pooja Singh. Discrete cosine transform and discrete fourier transform of rgb image. *International Journal of Computer Applications*, 2012.
- [38] Chitra Bhandari, Sumit Kumar, Sudha Chauhan, MA Rahman, Gaurav Sundaram, Rajib Kumar Jha, Shyam Sundar, AR Verma, and Yashvir Singh. Biomedical image encryption based on fractional discrete cosine transform with singular value decomposition and chaotic system. In *2019 International Conference on Computing, Power and Communication Technologies (GUCON)*, pages 520–523. IEEE, 2019.
- [39] Md Tahmid Hossain, Shyh Wei Teng, Dengsheng Zhang, Suryani Lim, and Guojun Lu. Distortion robust image classification using deep convolutional neural network with discrete cosine transform. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 659–663. IEEE, 2019.
- [40] Amnah Nasim, Agnese Sbröllini, Ilaria Marcantoni, Micaela Morettini, and Laura Burattini. Compressed segmented beat modulation method using discrete cosine transform. In *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 2273–2276. IEEE, 2019.
- [41] Ayodeji Olalekan Salau, Ifetayo Oluwafemi, Kehinde Felix Faleye, and Shruti Jain. Audio compression using a modified discrete cosine transform with temporal auditory masking. In *2019 International Conference on Signal Processing and Communication (ICSC)*, pages 135–142. IEEE, 2019.
- [42] Huazheng Wu, Xiangfeng Meng, Yurong Wang, Yongkai Yin, Xiulun Yang, Wenqi He, Guoyan Dong, and Hongyi Chen. High compressive ghost imaging method based on

- discrete cosine transform using weight coefficient matching. *Journal of Modern Optics*, 66(17):1736–1743, 2019.
- [43] Virendra P Vishwakarma. Fractional discrete cosine transform based approach for compensating the effect of light variations for robust face recognition. In *2018 Eleventh International Conference on Contemporary Computing (IC3)*, pages 1–6. IEEE, 2018.
- [44] Quoc Bao Dang, Kessi Louisa, Mickaël Coustaty, Muhammad Muzzamil Luqman, and Jean-Marc Ogier. A blind document image watermarking approach based on discrete wavelet transform and qr code embedding. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 8, pages 1–6. IEEE, 2019.
- [45] Amira Mofreh Ibraheem, Kamel Hussein Rahouma, and Hesham FA Hamed. Automatic mri breast tumor detection using discrete wavelet transform and support vector machines. In *2019 Novel Intelligent and Leading Emerging Sciences Conference (NILES)*, volume 1, pages 88–91. IEEE, 2019.
- [46] Gabriel de Alvarenga Ferreira and Tatiana Mariano Lessa Assis. A novel high impedance arcing fault detection based on the discrete wavelet transform for smart distribution grids. In *2019 IEEE PES Innovative Smart Grid Technologies Conference-Latin America (ISGT Latin America)*, pages 1–6. IEEE, 2019.
- [47] Ali Hadi Abdulwahid. A new concept of an intelligent protection system based on a discrete wavelet transform and neural network method for smart grids. In *2019 2nd International Conference of the IEEE Nigeria Computer Chapter (NigeriaComputConf)*, pages 1–6. IEEE, 2019.
- [48] Subrato Bharati, Mohammad Atikur Rahman, Sanjoy Mandal, and Prajoy Podder. Analysis of dwt, dct, bfo & pbfo algorithm for the purpose of medical image watermarking. In *2018*

- International Conference on Innovation in Engineering and Technology (ICIET)*, pages 1–6. IEEE, 2018.
- [49] A.G. Selvi J. and M.K. G. Probing image and video steganography based on discrete wavelet and discrete cosine transform. *2019 Fifth International Conference on Science Technology Engineering and Mathematics (ICONSTEM), Science Technology Engineering and Mathematics (ICONSTEM), 2019 Fifth International Conference on*, 1:21 – 24, 2019.
- [50] Virendra P Vishwakarma, Sahil Dalal, and Varsha Sisaudia. Efficient feature extraction using dwt-dct for robust face recognition under varying illuminations. In *2018 2nd IEEE International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES)*, pages 982–987. IEEE, 2018.
- [51] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.
- [52] Kishan Maladkar. 6 Types of Artificial Neural Networks Currently Being Used in ML, January 2018.
- [53] Saima Farhan, Muhammad Abuzar Fahiem, and Huma Tauseef. An ensemble-of-classifiers based approach for early diagnosis of alzheimer’s disease: classification using structural features of brain images. *Computational and mathematical methods in medicine*, 2014, 2014.
- [54] Xin Wei, Hui Wang, Bryan Scotney, and Huan Wan. Precise adjacent margin loss for deep face recognition. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 3641–3645. IEEE, 2019.

- [55] Dinesh Kumar et al. Performance evaluation of face recognition system using various distance classifiers. In *2018 Second International Conference on Computing Methodologies and Communication (ICCMC)*, pages 322–327. IEEE, 2018.
- [56] Rachid Ahdid, Khaddouj TAIFI, SAFI Said, and Bouzid MANAUT. Euclidean & geodesic distance between a facial feature points in two-dimensional face recognition system. *human-computer interaction*, 1:5, 2017.
- [57] DA Tarasov, AN Medvedev, AP Sergeev, AV Shichkin, and Alexander G Buevich. A hybrid method for assessment of soil pollutants spatial distribution. In *AIP Conference Proceedings*, volume 1863, page 050015. AIP Publishing, 2017.
- [58] CE Kiessling and Cyril J Tunis. Linearly separable codes for adaptive threshold networks. *IEEE Transactions on Electronic Computers*, 1(6):935–936, 1965.
- [59] Kdnji Nakayama, Yoshinori Kimura, and Hiroshi Katayama. Quantization level increase in human face images using multilayer neural network. In *Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan)*, volume 2, pages 1247–1250. IEEE, 1993.
- [60] Sudeep D Thepade and Deepa Abin. Face gender recognition using multi layer perceptron with otsu segmentation. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, pages 1–5. IEEE, 2018.
- [61] Yeon-Sik Ryu and Se-Young Oh. Automatic extraction of eye and mouth fields from a face image using eigenfeatures and ensemble networks. *Applied Intelligence*, 17(2):171–185, 2002.
- [62] R Vapenik, O Kainz, P Fecil’ak, and F Jakab. Human face detection in still image using multilayer perceptron solution based on neuroph framework. In *2016 international confer-*

- ence on emerging elearning technologies and applications (ICETA), pages 365–369. IEEE, 2016.
- [63] Alejandro Baldominos, Yago Saez, and Pedro Isasi. Evolutionary design of convolutional neural networks for human activity recognition in sensor-rich environments. *Sensors*, 18(4):1288, 2018.
- [64] Suleman Khan, M Hammad Javed, Ehtasham Ahmed, Syed AA Shah, and Syed Umaid Ali. Facial recognition using convolutional neural networks and implementation on smart glasses. In *2019 International Conference on Information Science and Communication Technology (ICISCT)*, pages 1–6. IEEE, 2019.
- [65] Saumya Kumar, Ravi M Vishwanath, SN Omkar, Abrar Majeedi, and Abhinandan Dogra. Disguised facial recognition using neural networks. In *2018 IEEE 3rd International Conference on Signal and Image Processing (ICSIP)*, pages 28–32. IEEE, 2018.
- [66] Hamed Habibi Aghdam and Elnaz Jahani Heravi. Guide to convolutional neural networks. *New York, NY: Springer. doi*, 10:978–3, 2017.
- [67] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [68] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [69] Gu Shengtao, Xu Chao, and Feng Bo. Facial expression recognition based on global and local feature fusion with cnns. In *2019 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, pages 1–5. IEEE, 2019.

- [70] Jing Li, Yang Mi, Gongfa Li, et al. Cnn-based facial expression recognition from annotated rgb-d images for human-robot interaction, 2019.
- [71] Ee Heng Chen, Philipp Röthig, Jöran Zeisler, and Darius Burschka. Investigating low level features in cnn for traffic sign detection and recognition. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 325–332. IEEE, 2019.
- [72] Jia Li and Zengfu Wang. Real-time traffic sign recognition based on efficient cnns in the wild. *IEEE Transactions on Intelligent Transportation Systems*, 20(3):975–984, 2018.
- [73] Chunpeng Wu, Wei Wen, Tariq Afzal, Yongmei Zhang, Yiran Chen, et al. A compact dnn: approaching googlenet-level accuracy of classification and domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5668–5677, 2017.
- [74] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.
<http://www.deeplearningbook.org>.
- [75] Kenneth O. Stanley. Neuroevolution: A different kind of deep learning, July 2017.
- [76] Seyed Mohammad Jafar Jalali, Parham M Kebria, Abbas Khosravi, Khaled Saleh, Darius Nahavandi, and Saeid Nahavandi. Optimal autonomous driving through deep imitation learning and neuroevolution. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pages 1215–1220. IEEE, 2019.
- [77] Filip Badan and Lukas Sekanina. Optimizing convolutional neural networks for embedded systems by means of neuroevolution. In *International Conference on Theory and Practice of Natural Computing*, pages 109–121. Springer, 2019.

- [78] Tian Zhang, Jia Wang, Yihang Dan, Yuxiang Lanqiu, Jian Dai, Xu Han, Xiaojuan Sun, and Kun Xu. Efficient training and design of photonic neural network through neuroevolution. *Optics Express*, 27(26):37150–37163, 2019.
- [79] Ahmed Aly, David Weikersdorfer, and Claire Delaunay. Optimizing deep neural networks with multiple search neuroevolution. *arXiv preprint arXiv:1901.05988*, 2019.
- [80] Xin Yao and Yong Liu. A new evolutionary system for evolving artificial neural networks. *IEEE transactions on neural networks*, 8(3):694–713, 1997.
- [81] Kenneth O Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127, 2002.
- [82] Yohannes Kassahun and Gerald Sommer. Efficient reinforcement learning through evolutionary acquisition of neural topologies. In *ESANN*, pages 259–266, 2005.
- [83] Roxana ZahediNasab and Hadis Mohseni. Neuroevolutionary based convolutional neural network with adaptive activation functions. *Neurocomputing*, 2019.
- [84] MG Sumithra and AK Devika. A study on feature extraction techniques for text independent speaker identification. In *2012 International Conference on Computer Communication and Informatics*, pages 1–5. IEEE, 2012.
- [85] Matthew A Turk and Alex P Pentland. Face recognition using eigenfaces. In *Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 586–587, 1991.
- [86] Yongwon Jeong and Hyung Soon Kim. New speaker adaptation method using 2-d pca. *IEEE Signal Processing Letters*, 17(2):193–196, 2009.

- [87] Rang MH Nguyen and Michael S Brown. Why you should forget luminance conversion and do something better. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6750–6758, 2017.
- [88] PAUL DANIEL Dumitru, MARIN Ploeanu, and DRAGOS Badea. Comparative study regarding the methods of interpolation. *Recent advances in geodesy and Geomatics engineering*, 1:45, 2013.
- [89] Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, 12(7):629–639, 1990.
- [90] Chourmouzios Tsotsios and Maria Petrou. On the choice of the parameters for anisotropic diffusion in image processing. *Pattern recognition*, 46(5):1369–1381, 2013.
- [91] Guillermo Sapiro. *Geometric partial differential equations and image analysis*. Cambridge university press, 2006.
- [92] Chenggen Quan, Yu Fu, and Hong Miao. Wavelet analysis of digital shearing speckle patterns with a temporal carrier. *Optics communications*, 260(1):97–104, 2006.
- [93] Marie Farge. Wavelet transforms and their applications to turbulence. *Annual review of fluid mechanics*, 24(1):395–458, 1992.
- [94] Mohammed Nabih Ali, EL-Sayed A El-Dahshan, and Ashraf H Yahia. Denoising of heart sound signals using discrete wavelet transform. *Circuits, Systems, and Signal Processing*, 36(11):4482–4497, 2017.
- [95] Fahima Tabassum, Md Imdadul Islam, and Mohamed Ruhul Amin. A simplified image compression technique based on haar wavelet transform. In *2015 International Conference*

- on Electrical Engineering and Information Communication Technology (ICEEICT)*, pages 1–9. IEEE, 2015.
- [96] Liu Chun-Lin. A tutorial of the wavelet transform. *NTUEE, Taiwan*, 2010.
- [97] Alessandro B Romeo, Cathy Horellou, and Jöran Bergh. A wavelet add-on code for new-generation n-body simulations and data de-noising (jofiluren). *Monthly Notices of the Royal Astronomical Society*, 354(4):1208–1222, 2004.
- [98] Jimmy Alexander Cortés-Osorio, Juan Bernardo Gómez-Mendoza, and Juan Carlos Riaño-Rojas. Velocity estimation from a single linear motion blurred image using discrete cosine transform. *IEEE Transactions on Instrumentation and Measurement*, 2018.
- [99] AM Raid, WM Khedr, Mohamed A El-Dosuky, and Wesam Ahmed. Jpeg image compression using discrete cosine transform-a survey. *arXiv preprint arXiv:1405.6147*, 2014.
- [100] Ziad M Hafeed and Martin D Levine. Face recognition using the discrete cosine transform. *International journal of computer vision*, 43(3):167–188, 2001.
- [101] Weilong Chen, Meng Joo Er, and Shiqian Wu. Illumination compensation and normalization for robust face recognition using discrete cosine transform in logarithm domain. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 36(2):458–466, 2006.
- [102] Sanjeev Pragada and Jayanthi Sivaswamy. Image denoising using matched biorthogonal wavelets. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 25–32. IEEE, 2008.
- [103] Xiu-Ge Zhu, Bao-Bin Li, and Deng-Feng Li. Orthogonal wavelet transform of signal based on complex b-spline bases. *International Journal of Wavelets, Multiresolution and Information Processing*, 10(06):1250054, 2012.

- [104] Alejandro Lopez-Rincon, Alberto Tonda, Mohamed Elati, Olivier Schwander, Benjamin Piwowarski, and Patrick Gallinari. Evolutionary optimization of convolutional neural networks for cancer mirna biomarkers classification. *Applied Soft Computing*, 65:91–100, 2018.
- [105] Junghwan Cho, Kyewook Lee, Ellie Shin, Garry Choy, and Synho Do. How much data is needed to train a medical image deep learning system to achieve necessary high accuracy? *arXiv preprint arXiv:1511.06348*, 2015.
- [106] Fully Connected Layers in Convolutional Neural Networks: The Complete Guide.
- [107] Musab COŞKUN, Özal YILDIRIM, Ayşegül UÇAR, and Yakup DEMİR. An overview of popular deep learning methods. *European Journal of Technic*, 7(2), 2017.
- [108] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.
- [109] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [110] Wu Junqin and Yu Junjun. An improved arithmetic of mfcc in speech recognition system. In *2011 International Conference on Electronics, Communications and Control (ICECC)*, pages 719–722. IEEE, 2011.
- [111] Pravinkumar Premakanthan and WB Mikhael. Speaker verification/recognition and the importance of selective feature extraction. In *Proceedings of the 44th IEEE 2001 Midwest Symposium on Circuits and Systems. MWSCAS 2001 (Cat. No. 01CH37257)*, volume 1, pages 57–61. IEEE, 2001.

- [112] Md Rabiul Islam, Md Fayzur Rahman, and Muhammad Abdul Goffar Khan. Improvement of speech enhancement techniques for robust speaker identification in noise. In *2009 12th International Conference on Computers and Information Technology*, pages 255–260. IEEE, 2009.
- [113] A Revathi and Y Venkataramani. Text independent composite speaker identification/verification using multiple features. In *2009 WRI World congress on computer science and information engineering*, volume 7, pages 257–261. IEEE, 2009.
- [114] Philippe Delsarte and Yves Genin. The split levinson algorithm. *IEEE transactions on acoustics, speech, and signal processing*, 34(3):470–478, 1986.
- [115] Richard J Mammone, Xiaoyu Zhang, and Ravi P Ramachandran. Robust speaker recognition: A feature-based approach. *IEEE signal processing magazine*, 13(5):58, 1996.
- [116] Homayoon Beigi. Speaker recognition: Advancements and challenges. *New trends and developments in biometrics*, pages 3–29, 2012.
- [117] Taif Alobaidi and Wasfy B Mikhael. Mixed nonorthogonal transforms representation for face recognition. *Circuits, Systems, and Signal Processing*, 38(4):1684–1694, 2019.
- [118] Meijing Li, Xiuming Yu, Keun Ho Ryu, Sanghyuk Lee, and Nipon Theera-Umpon. Face recognition technology development with gabor, pca and svm methodology under illumination normalization condition. *Cluster Computing*, 21(1):1117–1126, 2018.
- [119] Cheng-Yaw Low, Andrew Beng-Jin Teoh, and Cong-Jie Ng. Multi-fold gabor, pca, and ica filter convolution descriptor for face recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(1):115–129, 2017.
- [120] M Haq, Aamir Shahzad, Zahid Mahmood, A Shah, Nazeer Muhammad, and Tallha Akram. Boosting the face recognition performance of ensemble based lda for pose non-uniform

- illuminations and low-resolution images. *KSII Transactions on Internet and Information Systems*, 2019.
- [121] Muhammad Hameed Siddiqi, Rahman Ali, Adil Mehmood Khan, Young-Tack Park, and Sungyoung Lee. Human facial expression recognition using stepwise linear discriminant analysis and hidden conditional random fields. *IEEE Transactions on Image Processing*, 24(4):1386–1398, 2015.
 - [122] A. K. Sharma, U. Kumar, S. K. Gupta, U. Sharma, and S. L. Agrwal. A survey on feature extraction technique for facial expression recognition system. In *2018 4th International Conference on Computing Communication and Automation (ICCCA)*, pages 1–6, Dec 2018.
 - [123] Chengjun Liu and Harry Wechsler. Evolutionary pursuit and its application to face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(6):570–582, 2000.
 - [124] Ebenezer Owusu, Jamal-Deen Abdulai, and Yongzhao Zhan. Face detection based on multilayer feed-forward neural network and haar features. *Software: Practice and Experience*, 49(1):120–129, 2019.
 - [125] Michael A Nielsen. *Neural Networks and Deep Learning*. Determination Press, 2015.
 - [126] Ferdinando S Samaria and Andy C Harter. Parameterisation of a stochastic model for human face identification. In *Proceedings of 1994 IEEE workshop on applications of computer vision*, pages 138–142. IEEE, 1994.
 - [127] P Jonathon Phillips, Hyeonjoon Moon, Syed A Rizvi, and Patrick J Rauss. The feret evaluation methodology for face-recognition algorithms. *IEEE Transactions on pattern analysis and machine intelligence*, 22(10):1090–1104, 2000.

- [128] P Jonathon Phillips, Harry Wechsler, Jeffery Huang, and Patrick J Rauss. The feret database and evaluation procedure for face-recognition algorithms. *Image and vision computing*, 16(5):295–306, 1998.
- [129] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.
- [130] ImageNet. ImageNet.
- [131] ILSVRC. ImageNet Large Scale Visual Recognition Competition (ILSVRC).
- [132] Kunihiko Fukushima. Neocognitron. *Scholarpedia*, 2(1):1717, 2007.
- [133] Masakazu Matsugu, Katsuhiko Mori, Yusuke Mitari, and Yuji Kaneda. Subject independent facial expression recognition with robust face detection using a convolutional neural network. *Neural Networks*, 16(5-6):555–559, 2003.
- [134] Xiangyu Zhang, Jianhua Zou, Kaiming He, and Jian Sun. Accelerating very deep convolutional networks for classification and detection. *IEEE transactions on pattern analysis and machine intelligence*, 38(10):1943–1955, 2015.
- [135] Hamid Sadeghi and Abolghasem-A Raie. Human vision inspired feature extraction for facial expression recognition. *Multimedia Tools and Applications*, 78(21):30335–30353, 2019.
- [136] Ethem Alpaydin. Introduction to machine learning. [sl], 2010.
- [137] Yue-Fei Guo, Ting-Ting Shu, Jing-Yu Yang, and Shi-Jin Li. Feature extraction method based on the generalised fisher discriminant criterion and facial recognition. *Pattern Analysis & Applications*, 4(1):61–66, 2001.

- [138] Yu-Feng Yu, Dao-Qing Dai, Chuan-Xian Ren, and Ke-Kun Huang. Discriminative multi-layer illumination-robust feature extraction for face recognition. *Pattern Recognition*, 67:201–212, 2017.
- [139] Abdul Mannan, Kashif Javed, Seroosh Karim Noon, Haroon Atique Babri, et al. Optimized segmentation and multiscale emphasized feature extraction for traffic sign detection and recognition. *Journal of Intelligent & Fuzzy Systems*, 36(1):173–188, 2019.
- [140] Samira El Margae, Berraho Sanae, Ait Kerroum Mounir, and Fakhri Youssef. Traffic sign recognition based on multi-block lbp features using svm with normalization. In *2014 9th international conference on intelligent systems: theories and applications (SITA-14)*, pages 1–7. IEEE, 2014.
- [141] Samira El Margae, Sanae Berraho, Mounir Ait Kerroum, and Youssef Fakhri. Multi-stage fusion of local and global features based classification for traffic sign recognition. In *2014 International Conference on Multimedia Computing and Systems (ICMCS)*, pages 495–499. IEEE, 2014.
- [142] Huahui Yang, Chen Meng, and Cheng Wang. Data-driven feature extraction for analog circuit fault diagnosis using 1-d convolutional neural network. *IEEE Access*, 8:18305–18315, 2020.
- [143] Beicheng Ding and Penghui Chen. Hrrp feature extraction and recognition method of radar ground target using convolutional neural network. In *2019 International Conference on Electromagnetics in Advanced Applications (ICEAA)*, pages 0658–0661. IEEE, 2019.
- [144] Ling Zhang and Zhigang Zhu. Unsupervised feature learning for point cloud understanding by contrasting and clustering using graph convolutional neural networks. In *2019 International Conference on 3D Vision (3DV)*, pages 395–404. IEEE, 2019.

- [145] Mehmet Ali Kutlugün, Yahya Sirin, and MehmetAli Karakaya. The effects of augmented training dataset on performance of convolutional neural networks in face recognition system. In *2019 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 929–932. IEEE, 2019.
- [146] Xiangbang Meng, Yan Yan, Si Chen, and Hanzi Wang. A cascaded noise-robust deep cnn for face recognition. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 3487–3491. IEEE, 2019.
- [147] Abhinav Agrawal and Namita Mittal. Using cnn for facial expression recognition: a study of the effects of kernel size and number of filters on accuracy. *The Visual Computer*, 36(2):405–412, 2020.
- [148] Sungick Kong, Jonghee Park, Sang-Seol Lee, and Sung-Joon Jang. Lightweight traffic sign recognition algorithm based on cascaded cnn. *International Conference on Control Robot Systems*, pages 506–509, 2019.
- [149] Vijini Mallawaarachchi. Introduction to Genetic Algorithms — Including Example Code, November 2019.
- [150] Kuntal Kumar Pal and KS Sudeep. Preprocessing for image classification by convolutional neural networks. In *2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, pages 1778–1781. IEEE, 2016.
- [151] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient back-prop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.
- [152] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.

- [153] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.
- [154] Anil K Jain and Stan Z Li. *Handbook of face recognition*. Springer, 2011.
- [155] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks*, 32:323–332, 2012.
- [156] Zhe Zhu, Dun Liang, Songhai Zhang, Xiaolei Huang, Baoli Li, and Shimin Hu. Traffic-sign detection and classification in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2110–2118, 2016.
- [157] Matthew Stewart Researcher, PhD. Advanced Topics in Deep Convolutional Neural Networks, January 2020. Library Catalog: towardsdatascience.com.
- [158] Dominik Jelšovka, Róbert Hudec, and Martin Brežňan. Face recognition on feret face database using lda and cca methods. In *Telecommunications and Signal Processing (TSP), 2011 34th International Conference on*, pages 570–574. IEEE, 2011.
- [159] J. Chen and W. K. Jenkins. Facial recognition with pca and machine learning methods. In *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 973–976, Aug 2017.
- [160] Mustafa Zuhaer Nayef Al-Dabagh, Mustafa H Mohammed Alhabib, and Firas H AL-Mukhtar. Face recognition system based on kernel discriminant analysis, k-nearest neighbor and support vector machine. *International Journal of Research and Engineering*, 5(3):335–338, 2018.

- [161] Stéphane Mallat. *A wavelet tour of signal processing*. Elsevier, 1999.
- [162] Shilpashree Rao and MV Bhaskara Rao. A novel triangular dct feature extraction for enhanced face recognition. In *Intelligent Systems and Control (ISCO), 2016 10th International Conference on*, pages 1–6. IEEE, 2016.
- [163] Arun Ramaswamy and WB Mikhael. Multitransform/multidimensional signal representation. In *Circuits and Systems, 1993., Proceedings of the 36th Midwest Symposium on*, pages 1255–1258. IEEE, 1993.
- [164] Sung-Kwun Oh, Sung-Hoon Yoo, and Witold Pedrycz. A comparative study of feature extraction methods and their application to p-rbf nns in face recognition problem. *Fuzzy Sets and Systems*, 305:131–148, 2016.
- [165] Jou Lin and Ching-Te Chiu. Low-complexity face recognition using contour-based binary descriptor. *IET Image Processing*, 11(12):1179–1187, 2017.
- [166] Junjie Chen, Xiren Miao, Hao Jiang, Jing Chen, and Xinyu Liu. Identification of autonomous landing sign for unmanned aerial vehicle based on faster regions with convolutional neural network. In *Chinese Automation Congress (CAC), 2017*, pages 2109–2114. IEEE, 2017.
- [167] Zhiyin Cai, Wei Gao, Zhuliang Yu, Jinhong Huang, and Zhaoquan Cai. Feature extraction with triplet convolutional neural network for content-based image retrieval. In *Industrial Electronics and Applications (ICIEA), 2017 12th IEEE Conference on*, pages 337–342. IEEE, 2017.
- [168] Mukul V Shirvaikar. An optimal measure for camera focus and exposure. In *System Theory, 2004. Proceedings of the Thirty-Sixth Southeastern Symposium on*, pages 472–475. IEEE, 2004.

- [169] Yi Yang, Hengliang Luo, Huarong Xu, and Fuchao Wu. Towards real-time traffic sign detection and classification. *IEEE Transactions on Intelligent transportation systems*, 17(7):2022–2031, 2015.
- [170] Xudong Jiang. Feature extraction for image recognition and computer vision. In *2009 2nd IEEE International Conference on Computer Science and Information Technology*, pages 1–15. IEEE, 2009.
- [171] Oleksandr Oksiiuk, Liudmyla Tereikovska, and Ihor Tereikovskiy. Determination of expected output signals of the neural network model intended for image recognition. In *2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*, pages 596–599. IEEE, 2017.
- [172] Myroslav Komar, Pavlo Yakobchuk, Vladimir Golovko, Vitaliy Dorosh, and Anatoliy Sachenko. Deep neural network for image recognition based on the caffe framework. In *2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP)*, pages 102–106. IEEE, 2018.
- [173] Zhe Zhu, Dun Liang, Songhai Zhang, Xiaolei Huang, Baoli Li, and Shimin Hu. Traffic-sign detection and classification in the wild. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [174] Georgios Papanikas. Real-time automatic transcription of drums music tracks on an fpga platform. *Master’s Thesis*, 2012.
- [175] HB Kekre, Vaishali Kulkarni, Prashant Gaikar, and Nishant Gupta. Speaker identification using spectrograms of varying frame sizes. *International Journal of Computer Applications*, 50(20), 2012.

- [176] John S Garofolo, Lori F Lamel, William M Fisher, Jonathan G Fiscus, and David S Pallett. Getting started with the darpa timit cd-rom: An acoustic phonetic continuous speech database. *National Institute of Standards and Technology (NIST), Gaithersburgh, MD*, 107:16, 1988.
- [177] Pawan Kumar, Nitika Jakhanwal, and Mahesh Chandra. Text dependent speaker identification in noisy environment. In *2011 International Conference on Devices and Communications (ICDeCom)*, pages 1–4. IEEE, 2011.
- [178] Ronald A Cole, Mike Noel, and Victoria Noel. The cslu speaker recognition corpus. In *Fifth International Conference on Spoken Language Processing*, 1998.
- [179] Md Afzal Hossan, Sheeraz Memon, and Mark A Gregory. A novel approach for mfcc feature extraction. In *2010 4th International Conference on Signal Processing and Communication Systems*, pages 1–5. IEEE, 2010.
- [180] Sangeeta Biswas, Shamim Ahmad, and Md Khademul Islam Mollad. Speaker identification using cepstral based features and discrete hidden markov model. In *2007 International Conference on Information and Communication Technology*, pages 303–306. IEEE, 2007.
- [181] Tilo Schurer. An experimental comparison of different feature extraction and classification methods for telephone speech. In *Proceedings of 2nd IEEE Workshop on Interactive Voice Technology for Telecommunications Applications*, pages 93–96. IEEE, 1994.