Emotion recognition using facial feature extraction

Demiyan Smirnov
Rowan University

EMOTION RECOGNITION USING FACIAL FEATURE EXTRACTION

by

Demiyan Smirnov

A Thesis

Submitted to the
Department of Electrical and Computer Engineering
College of Engineering
In partial fulfillment of the requirement
For the degree of
Master of Science in Electrical Engineering
at
Rowan University
April 19, 2019

Thesis Chair: Ravi Ramachandran, Ph.D.

Dedication

This work is dedicated to the two women who mean the most to me: my persistent mother, Sima Smirnova, and my loving grandmother, Faina Lebedeva.

Acknowledgement

The research presented in this thesis has been made possible by several key factors. Grants from the National Science Foundation were a large contributor. Next, the idea for this research would not have come about without Dr. Ravi Ramachandran of Rowan University and Dr. Rajani Muraleedharan of Saginaw Valley State University. They provided much needed continuous support through their suggestions of different approaches and through there insight on the best methods to assess accuracy and present results. Rowan University (their software suite, in particular) is also worth acknowledging as a benefactor for this research.

The support of family and friends could also not be understated. They helped in creating a positive atmosphere and driving motivation for completing the research and this written thesis.

Abstract

Demiyan Smirnov
EMOTION RECOGNITION USING FACIAL FEATURE EXTRACTION
2013-2018
Ravi Ramachandran, Ph.D.
Master of Science in Electrical Engineering

Computerized emotion recognition systems can be powerful tools to help solve problems in a wide range of fields including education, healthcare, and marketing. Existing systems use digital images or live video to track facial expressions on a person's face and deduce that person's emotional state. The research presented in this thesis explores combinations of several facial feature extraction techniques with different classifier algorithms. Namely, the feature extraction techniques used in this research were Discrete Cosine/Sine Transforms, Fast Walsh-Hadamard Transform, Principle Component Analysis, and a novel method called XPoint. Features were extracted from both global (using the entire facial image) and local (using only facial regions like the mouth or eyes) contexts and classified with Linear Discriminant Analysis and k-Nearest Neighbor algorithms. Some experiments also fused many of these features into one system in an effort to create even more accurate systems.

The system accuracy for each feature extraction method/classifier combination was calculated and discussed. The combinations that performed the best produced systems between 85%-90% accurate. The most accurate systems utilized Discrete Sine Transform from global and local features in a Linear Discriminant Analysis classifier, as well as feature fusion of all features in a Linear Discriminant Classifier.

Table of Contents

Abstract ii

List of Figures vii

List of Tables ix

Chapter 1: Introduction 1

Problem Statement 1

Hypothesis 1

Significance of Research 2

Research Approach 3

Literature Survey 3

Applied Software 4

Feature Extraction 5

Classification 6

Accuracy Assessment 6

Thesis Structure 7

Chapter 2: Literature Survey 8

Emotion: What is it? 8

Universality of vs. Cultural Influence on Facial Expressions 9

Varied Approaches to Emotion Recognition 13

Principle Component Analysis 14

Discrete Cosine Transform 21

Fast Walsh-Hadamard Transform 24

Facial Action Coding System 26

Table of Contents (Continued)

Chapter 3: Methods and Experimental Procedure 29

Basic Approach to Emotion Recognition 29

Facial Image Database 31

Cohn-Kanade Database 32

JAFFE Database 34

Facial Detection 35

Synthetic Minority Oversampling Technique (SMOTE) 38

Preprocessing 40

Scaling 41

Patching 41

Normalization 42

Energy Thresholding 43

Feature Extraction 45

DCT/DST 45

Principle Component Analysis 47

X-Point Method 49

Automated Point Placement on Facial Landmarks 51

ASM Fitter 53

ASM Molder 54

ASM Searcher 56

Classification 59

Table of Contents (Continued)

Linear Discriminant Analysis 59

Kth Nearest Neighbor 63

Experimental Procedure 64

Data Partitioning 65

Top-N Ranking 65

Feature Fusion 67

Calculating System Accuracy 67

Chapter 4: Results and Discussion 70

Basic Approach to Results and Discussion 70

Cohn-Kanade Dataset Results 71

Case for Viola-Jones 71

Case for Shrunken Images 74

Case for SMOTE 76

Further Increasing the Natural Sample Sizes for Each Emotion 79

Frequency Features with KNN Classifier 83

Further Understanding of Accuracy 84

Case for Local Features 87

Case for 20-Point Method 89

Best Performing Features using Cohn-Kanade 90

PCA Results 92

JAFFE Results 94

Table of Contents (Continued)

Automatic Facial Point Annotator 96

Comparisons to Cohn-Kanade 97

Best Performing Features Using JAFFE..100

Fusion Results..102

Chapter 5: Conclusion..106

Summary of Findings..106

Recommendations and Feasibility 107

Future Work 108

References...110

List of Figures

Figure Page

Figure 2.1. The modes of variation (eigenfaces) of the Cohn-Kanade dataset................18

Figure 2.2. A reconstruction of a test image using images from a training set 19

Figure 2.3. A sample transformation matrix after a 2D DCT was applied to an image..22

Figure 2.4. The process of collecting 2D DCT coefficients to form a 1D DCT feature vector 24

Figure 2.5. Walsh Matrices of different sizes...25

Figure 3.1. 68 landmark points used in Extended Cohn-Kanade database......................33

Figure 3.2. Feature types used by Viola-Jones Algorithm..37

Figure 3.3. Viola-Jones algorithm cropping out faces in an image 38

Figure 3.4. Examples for facial patches..42

Figure 3.5. Effect of normalization on an image 43

Figure 3.6. High energy vs. low energy image 44

Figure 3.7. All attempted configurations of the X-Point method 50

Figure 3.8. Typical confusion matrix..69

Figure 4.1. Initial set of experiments run on the original Cohn-Kanade Database using an LDA Classifier...70

Figure 4.2. Set of experiments detailing results with Viola-Jones applied to each input image..72

Figure 4.3. Set of experiments detailing results with facial images scaled to 128 x 128 pixels..74

Figure 4.4. Set of experiments detailing results with facial images scaled to 64 x 64 pixels..75

List of Figures (Continued)

Figure Page

Figure 4.5. Set of experiments detailing results with full SMOTE 76

Figure 4.6. Set of experiments detailing results with half SMOTE................................78

Figure 4.7. Set of experiments detailing results with extra samples...............................80

Figure 4.8. Set of experiments detailing results with extra samples...............................81

Figure 4.9. Set of experiments detailing results with DST features of length 120...83

Figure 4.10. Experiments detailing the results of using the Top N class scores for classification rather than Top 1st 84

Figure 4.11. Set of experiments detailing the results of using the Top N class scores for classification rather than Top 1st 85

Figure 4.12. Set of experiments detailing the results of using the Top N class scores for classification rather than Top 1st with extra samples 86

Figure 4.13. Set of experiments detailing the results of different XPoint features in a kNN Classifier...89

Figure 4.14. Class averages for best performing features from Cohn-Kanade dataset 91

Figure 4.15. System accuracies for various feature extracted from JAFFE database, without using SMOTE 98

Figure 4.16. System accuracies for various feature extracted from JAFFE database, with SMOTE...99

Figure 4.17. Class averages for best performing features from the JAFFE dataset 100

Figure 4.18. Class accuracies for Cohn-Kanade feature fusion experiments 103

Figure 4.19. Class accuracies for JAFFE feature fusion experiments 104

List of Tables

Table Page

Table 2.1. Emotions annotated using the Facial Action Coding System.........................27

Table 3.1. Breakdown in sample size for each emotion in Cohn-Kanade database...34

Table 3.2. Breakdown in sample size for each emotion in JAFFE database 35

Table 4.1. Breakdown in sample size after Cohn-Kanade Database was processed with Viola-Jones..73

Table 4.2. Breakdown in sample size for each emotion after full SMOTE is applied..77

Table 4.3. Breakdown in sample size for each emotion after half SMOTE is applied..78

Table 4.4. Breakdown in sample size for each emotion after classification revision and extra samples were added..79

Table 4.5. Set of experiments detailing the LDA system accuracies for global and local features with all data provisions (Viola-Jones, 64x64 images, SMOTE, and extra samples 88

Table 4.6. System accuracies for best feature extracted from Cohn-Kanade database...91

Table 4.7. Set of experiments detailing the results of PCA features and an LDA classifier..92

Table 4.8. Confusion matrix for a PCA feature vector of length 200 in an LDA classifier..94

Table 4.9. Set of experiments detailing the results of PCA features and an kNN classifier 94

Table 4.10. Emotion sample size distributions for the JAFFE database 95

Table 4.11. Accuracy of placed points from automated tool compared to Cohn-Kanade ground truth 96

Table 4.12. System accuracies for best feature extracted from JAFFE database 101

List of Tables (Continued)

Table Page

Table 4.13. System accuracies for Cohn-Kanade feature fusion experiments 103

Table 4.14. System accuracies for JAFFE feature fusion experiments 104

Chapter 1

Introduction

Problem Statement

The concept of emotion has a place in evolutionary history and it even predates the rise of human beings (Darwin, 1899). This piece of knowledge corroborates the idea that some emotions are innate in humans, rather than just socially constructed (Ekman, 1998). While processing emotions may come naturally to most people, computers, on the other hand, have struggled with the execution of this concept for decades. Much research has been gathered in fields like computer vision and machine learning in the effort to use computers to accurately classify emotions portrayed by a human. This effort includes analyzing human speech, gestures, and facial expressions. The work presented here will address one of these challenges. Specifically, this study aims to answer the following question: How can the principles of engineering and mathematics be applied to create a system that can recognize emotions in given facial images (with over 90% accuracy)?

Hypothesis

1. Using simple features (Discrete Sine/Cosine Transform coefficients) and classifiers (Linear Discriminant Analysis and k Nearest Neighbor) can lead to a reasonably accurate facial emotion recognition system.
2. Breaking up the facial image into frames/patches and selecting specific frames/patches based on certain criteria will positively impact the system accuracy.
3. A sub-system for automated point placement on facial landmarks can benefit from a frequency-based search criterion.

4. A simplified version of the Facial Action Coding System can be used to extract features from a facial image by taking distances between key points on the face.
5. Fusing multiple feature sets and classifiers into one system will produce a comparably more accurate system.

Significance of Research

Many areas offer potential for the application of automated human facial expression recognition. Such fields include education, marketing, security, and medicine. Essentially, wherever a human is present to evaluate emotion, a computer can also aid in the analysis. In an educational setting, emotion recognition can be applied to classrooms to gauge student interest in learning material (Saneiro, Santos, Salmeron-Majadas, & Boticario, 2014). If the faces of students start to show disinterest or frustration, a computer system can alert the teacher to reconsider the teaching approach. Similarly, in marketing, advertisement agencies can present a participant with commercials and track their emotional states to see if the presented materials elicit the desired responses (Shergill, Diegel, Sarrafzadeh, & Shekar, 2008). Live monitoring of people's expressions can also provide vital security information. If people are exhibiting fear or disgust, an alert generated by a computerized security system informs a responder of the event (Butalia, Ingle, & Kulkarni, 2012). Lastly, an emotion recognition system can aid medical professionals as well. The system can be part of a tele-presence package that informs a doctor or psychologist of a patient's mental state. When something is detected as going awry, the professional can change their approach to helping the patient.

The initial motivation for this research was the development of a tool to help acclimate autistic children to processing emotion. Many people with autism struggle when expressing their own emotions and have a difficult time understanding the emotions of others (Losh & Capps, 2006). A computer-based emotion recognition tool would feature small puzzles for autistic children that would prompt them to mimic different expressive faces during varied scenarios. The tool would track their expressions and classify their emotions as correct or incorrect, given the prompt. Ultimately, these exercises would condition the children to understand what kind of responses are typically appropriate in certain situations.

Research Approach

In order to create the emotion recognition system, an iterative approach was taken. Many experiments were performed with varying parameters and the results were recorded and compared. An emphasis was placed on trial and error, where changes that contributed positively to system accuracy were adopted for future iterations. The system was built gradually, with each parameter refined to an optimal value. The system evaluated seven emotions (neutral, angry, disgusted, fearful, happy, sad, and surprised) from two data sets (Extended Cohn-Kanade and JAFFE). The two datasets were not mixed in any experiments, only one was focused on at a time.

Literature Survey. Prior to starting any system development, topics central to emotion recognition were examined, including the concept of emotions (including how they are presented through facial expression) and the different types of approaches that have already been attempted in the effort to solve the problem of automated emotion recognition.

With insight on human emotion, the classifications of each emotion became clearer. More specifically, the literature survey revealed a deliberate methodology to annotate facial movements for various regions on the face (like brows, eyes, and mouth) and correlate them to emotion (Ekman, Friesen, & Tomkins, Facial Affect Scoring Technique: A First Validity Study, 1971). Each methodology had direct implications on the developed emotion recognition system because the system took specific regions of interest into account.

Learning about prior approaches in automated emotion recognition aided in two respects. First, providing a landscape of methods that were already attempted aided in avoiding the occurrence of repeat experiments. In other words, research of the prior feature extraction and classification combinations guaranteed that the work presented in this thesis was novel. Second, the surveyed literature provided insight on approaches that had not been considered before. While more extensive information on alternate methods will be presented later in this document, the leap from global facial features to local ones was a decision made due to papers that explained the benefits of a system using local features.

Applied Software. After establishing the current state of automated emotion recognition through the literature survey, the platform on which to build an emotion recognition system was needed. MATLAB was chosen for several reasons. The platform's scripting language allows for fast prototyping, offering extensive libraries of pre-written and well-documented signal processing functions which are useful in the creation of the proposed system. Therefore, MATLAB was properly equipped to handle the entire end-to-end flow of the emotion recognition system. Tasks performed in MATLAB included reading in the image files, isolating the face from the rest of the image, pre-processing the face image, performing feature extraction, classifying the emotion, and assessing the system accuracy. A system like this could have also been written in Python (which is also a scripting language with pre-written libraries of functions) or in a C language using image processing libraries like OpenCV, but MATLAB was chosen based on the benefits of fast prototyping, the program's library of well documented functions, and prior experience with MATLAB.

The scope of this project also encouraged the selection of a software like MATLAB. The emotional recognition system was created primarily for exploring different feature extraction methods and classifiers. Rather than designing a user-friendly product, the intention was to determine frameworks and potential algorithms that could possibly be used for different applications in the future. As a result, creating graphical user interface elements or easy-to-use executable files was not a priority. Moreover, since there was no expected end-user for this particular research, there were also no specifications that such elements could be tailored to. Much of the code exists as scripts with hardcoded variables that need to be manually run with MATLAB. Ultimately, the research did not need to 'look nice' for other people to physically use and instead, only needed to exist to prove image processing concepts. As a result, the prototypical and scripting nature of MATLAB was a good fit for this research.

Feature Extraction. To a computer, an image is a signal. The pixel values of the image determine how bright each pixel is displayed. As with any signal, there are a multitude of ways to interpret and manipulate it to learn more about it. The process of feature extraction, in the context of the emotion recognition system, was used to normalize each image into a format where hidden patterns could emerge. Throughout the process of developing the system, different feature extraction techniques were applied on facial images in order to determine which algorithms produced the most valuable information. Techniques included frequency domain transforms, principle component analysis, and biological geometry analysis.

Classification. Extracted features acted as input for a classification system. The purpose of classification is to compare an unknown entity with groups of known entities, and ultimately assign the entity in question to one of those groups. Technically speaking, the groups are called classes. In this case, the unknown and known entities were facial images, and the classes were emotions. Each emotion had a collection of corresponding facial images in a format dependent on the type of features that were extracted. Classifiers (such as Linear Discriminant Analysis, k-Nearest Neighbor, Neural Networks, and Vector Quantization) created mathematical models of each emotion class using the library of known extracted features. The mathematic models generated from these classifiers could then take features from new facial images, perform classifier-dependent functions, and output a label for the emotion that the input features most closely resembled.

Accuracy Assessment. With so many varying parameters, such as different feature extraction algorithms and classifiers, organizing all of the results was imperative. Extensive accuracy assessments were carried out after each experiment. The results of these assessments influenced the course of the work in the development of the system. If a particular combination of feature and classifier were producing more accurate results, they were explored further. To determine how accurate a trial was, the correct number of classifications was divided by the total number of classifications. For a trial, each emotion had its own 'class accuracy' and these values could be averaged together to get a 'trial accuracy.’ Since multiple trials were run, the trial accuracies could be further averaged to get an overall ‘system accuracy.’ This final value was used when determining if the feature/classifier combination was successful. The specifics of system accuracy are detailed further in Chapter 3.

Thesis Structure

This thesis is divided into 5 main parts, which are the chapters of this work. In this chapter, some background is provided on the concept of emotion and the rest provides information on previously explored methods for facial emotion classification. After this introduction (Chapter 1), the literature survey is further detailed in Chapter 2. Chapter 3 is entitled Methods and Experimental Procedure. With the basic concepts established in the prior chapter, specific application of these concepts in this research is explained. All of the different feature extraction methods used are explained and the way that experiments were conducted is also discussed. The last two chapters are: Results and Discussion, and the Conclusion. The detailed outcomes from each experiment are presented in Chapter 4 along with in-depth implications and explanations. In Chapter 5 the claims from the Hypothesis are compared to the actual results and some possibilities for future work are provided as well.

Chapter 2

Literature Survey

Emotion: What is it?

As with most topics being studied, an appropriate place to start is with a definition of terms. In this thesis, the primary term in question is: emotion. Naturally, the first place to look would be a dictionary, and Merriam-Webster's entry for the word emotion is as follows: "the affective aspect of consciousness, a state of feeling" (Merriam-Webster, 2015). At their core, emotions are feelings. They are reactions and states of being that people experience every day. The prolific psychologist Dr. Paul Ekman, who has spent most of his career understanding the physical and social nuances of emotion, attributes the function of emotion to mobilizing an organism to deal quickly with important interpersonal encounters. Moreover, emotions play a role when people are around others as much as when they are alone (Ekman, Basic Emotions, 1999). Emotions are elicited by triggers in our environment and serve as a way to acknowledge and compartmentalize those triggers.

A common method of cataloging the wide array of emotions that people feel is by placing those emotions on a 2D plot, with pleasantness on one axis and severity on the other. An emotion like anger would be rated as unpleasant and quite severe while boredom would be close to the center: neither pleasant/unpleasant nor severe. According to Dr. Ekman, there is a set of 7 basic emotions which then produce the rest of the spectrum of possible emotions, either through composites or ranging severities (Ekman, Afterword: Universality of Emotional Expression? A Personal History of the Dispute, 1998). The seven basic emotions are neutral, angry, disgusted, fearful, happy, sad, and surprised.

The scope of this research concentrates on one way that emotion can be manifested physically in the human body: through facial expressions. The human face features 42 muscles (Jack, Garrod, & Schyns, 2014), which can be activated voluntarily or instinctively. As with other parts of the body, the face is capable of making gestures. This includes sticking out a tongue or winking. However, these voluntary actions of the face were not of interest. Rather, the system was developed in order to analyze the instinctive responses that the face makes when an emotion is triggered.

Universality of vs. Cultural Influence on Facial Expressions. A long debate exists on the topic of facial emotion expression. Are the faces that people make as they emote innate in all humans or are facial expressions exclusively something people learn through their culture? The first scientifically significant answer to this question came from Charles Darwin, the progenitor of the idea of evolution through natural selection. Several years after he wrote his well-known work entitled Origin of Species, he produced another work entitled The Expression of the Emotions in Man and Animals in 1872. In the latter, he postulated that emotion had an evolutionary basis and therefore facial expressions were programmed into human beings from the beginning of the species. However, this theory soon came under harsh scrutiny as cultural relativists began to make their counter-claims (Ekman, Afterword: Universality of Emotional Expression? A Personal History of the Dispute, 1998).

The backlash to Darwin's ideas came during a time when the concept of an Aryan Race in Nazi Germany and the practice of eugenics emerged as dangerous movements that threatened the world at large. These ideologies claimed that the innate differences in humans were cause for discrimination amongst people, since some qualities were viewed as superior to others. The resulting opposition and repulsion to these ideas became so strong that it caused the scientific community to heavily favor nurture in the “nature versus nurture” debate. Any acknowledgment of universal traits in humans was feared to lead to racism (Ekman, Afterword: Universality of Emotional Expression? A Personal History of the Dispute, 1998).

In the facial expression debate, specifically, prominent cultural relativists included Margaret Mead and Ray Birdwhistell. To cultural relativists like them, all facial expressions that were used to display emotions were learned from the surrounding society and did not arise without deliberate effort. Any similarities in facial expression responses (like smiling to portray happiness or frowning when sad) were voluntary consequences and not hardwired responses (Ekman, Afterword: Universality of Emotional Expression? A Personal History of the Dispute, 1998). These ideas presented a rejection of Darwin’s theory of universal expressions. Much of the evidence for cultural relativism in terms of facial emotion came from the idea that there was no accidental expression, only communication between two people. All the different contortions of the face were a method of non-verbal communication that were deliberately enacted by the individual. This idea was the founding principle behind kinesics, which was a theory created by Birdwhistell in 1952, and was the leading explanation of human emotion at the time (Birdwhistell, 1952).

In 1965, Paul Ekman was a young psychologist that entered the scene with the hope of settling the debate once and for all. In his approach, he used carefully constructed experiments to acquire tangible evidence. He did not have a specific claim (universal or culture relativism) before starting his research; he was merely aiming to let evidence lead him to a conclusion. Moreover, he would not let trends or fear of being labeled as a racist skew his findings. For the next five years he devised several experiments that helped shape his most influential hypothesis on the matter (Ekman, Basic Emotions, 1999).

One of Ekman's experiment involved visiting pre-literate tribes in New Guinea and telling them a story. Afterwards, tribe members were given three images of facial expressions, from which they had to choose a facial expression that was most relevant to the story. The accuracy with which the people picked emotions were important data in confirming ideas about universal expressions. The pre-literacy state of the subjects was a control for the experiment because this ensured the tribe members did not pass along ideas of culturally acceptable expressions through mass media.

In another experiment, a control was set in place that isolated subjects as they watched two films. In this case, the participants could not be influenced to emote differently by people around them. The first film was meant to elicit disgust, fear, and sadness while the other was a neutral film. Using a system that will be described in Section 2.2.4, the facial responses of the subjects were annotated and ultimately lead to the finding that the subjects displayed universal reactions. Since the group of people consisted of remotely different cultures, American and Japanese, the case for universality was further substantiated (Matsumoto, 1991).

Through research, facial expression were determined to arise from a combination of universal and cultural factors. There are two influences related to a facial expression manifesting from an emotion.

Focusing on universal factors that all human beings share, there is an innate response that triggers the muscles in the face when a basic emotion is evoked. Perhaps the most relevant evidence that facial expressions are hardwired into humans is that even blind people, who were born blind, show the same expressions for the same emotions. There is no way that they could have learned the facial expressions from their culture because it would not have been visually communicated to them (News Editor, 2015). The universal expressions that Paul Ekman identified were amusement, anger, contempt, contentment, disgust, embarrassment, excitement, fear, guilt, pride, relief, sadness, satisfaction, pleasure, and shame (Ekman, Basic Emotions, 1999). A shorter version of this list highlights the emotions evaluated in this research: anger, disgust, fear, happiness, sadness, and surprise. From an evolutionary standpoint, universal outside markers for unseen inner thoughts provide human beings with many benefits.

On the other hand, there is also a strong cultural element that influences the way a face can be shaped when expressing emotion. For example, a number of ‘display rules’ can be taught to people to alter their expressions. Display rules include: exaggerating or toning down an expression, neutralizing an expression by forcing the muscles back to a neutral pose, reacting to an emotion with another emotion (e.g., getting angry and then feeling disgusted by the initial anger response), and reacting to an elicitor with two emotions at once. In all of these cases, the face will create an emotion that can be considered a ‘blend.’ Blends are harder to recognize across cultures because different cultures may have different display rules. An example of a situation that can be interpreted in different ways by different cultures involves reacting to a coworker’s promotion. In some cultures, sadness can freely be displayed on the face as a sign of defeat while other cultures may encourage showing happiness for the coworkers. In this second case, a composite expression could be displayed and may not be universally recognized because it would contain a unique blend of sadness masked by happiness (Ekman, Universals and Cultural Differences in Facial Expressions of Emotion, 1971).

Display rules were the biggest obstacle that researchers had to overcome when navigating the topic of universality of facial expressions. When the rules were not taken into account, all expressions appeared relatively different with no consensus on what universal expressions would look like. Several conditions had to be met in order to dig out the true universal expressions from underneath the avalanche of display rules. First, it was important to isolate subjects when capturing their emotions. Since display rules are a cultural influence, subjects are more likely to use them in company. Next, it was important to use a high speed camera (with a high frame rate) to capture the entire sequence of emotion. As a result, the initial primal response could be captured first, followed by the subsequent triggering of display rules. Lastly, rather than asking the subjects to pose an expression, their responses were elicited naturally as a response to videos that were shown to them. These three factors finally allowed Dr. Ekman to conclusively confirm the existence of universal facial expression (Ekman, Afterword: Universality of Emotional Expression? A Personal History of the Dispute, 1998).

Varied Approaches to Emotion Recognition

Prior to starting the literature survey, universality of facial expressions was adopted as an underlying assumption. For the emotion recognition system, such an assumption would not necessarily be helpful or harmful. The system is flexible enough to be adapted for use with specific cultures if their expressions are different from other cultures. An emotion recognition system is a highly variable construction and the literature survey further revealed the wide diversity of techniques that can be implemented.

Multiple paradigms exist when it comes to emotion recognition approaches. Some methods involve holistic systems that focus solely on facial musculature while other methods involve mathematical models with predictive capabilities. Further still, systems can examine the full face (a global feature), separate parts of the face (local features), or even a mixture of the former and the latter. The modular nature of the emotion recognition problem leads to an assortment of solutions, each striving to achieve an accuracy that rivals or surpasses human proficiency.

Principle Component Analysis. During the literature survey, Principle Component Analysis (PCA) was found to be a process used in many emotion recognition systems. In applications relating to facial images, PCA can also be correlated with ‘eigenface’ methods. PCA is a dimensionality reduction tool that can find common modes of variation amongst multiple input signals (Principle Component Analysis, 2015). These modes can be referred to as ‘eigenvectors,’ and when the input signals are facial images, the term eignfaces serves as a clever way of describing the modes. Several factors are interesting to note about PCA: the resulting lowered dimensionality cuts down on processing time, eigenfaces can contain many variations in one mode which reduces the need to do analysis on segments of the image (Paithane, Hullyalkar, Behera, Sonakul, & Manmode, 2014), and PCA has even been shown to mimic recognition behaviors of the human brain. The latter item has been demonstrated in a number of papers and is particularly beneficial in respect to two areas: PCA performs better when faces with more distinct features are classified (and by extension, with caricaturizations of a face) and PCA classifies more accurately intra-racial faces (from the same race) rather than inter-racial faces (from different races) (Calder, Burton, Miller, Young, & Akamatsu, 2001).

In order to get a better understanding of PCA, explaining how PCA is performed can be helpful. The first required ingredient for PCA is an Input Data Matrix (IDM). Each row of the matrix represents an instance of a signal in this matrix. An instance of a signal is represented by all of the pixel values of an image condensed down to a single long vector. Typically, each row of the image can be scanned and concatenated to the resultant vector. The Input Data Matrix takes in all of the images, regardless of what their classification is (i.e., regardless of which emotion is being displayed). Lastly, when the Input Data Matrix is built, the mean is taken and subtracted from each row. Imagining that each row is a point in n-space (n = number of columns, or vector size), subtracting the mean from the entire set essentially centers each point about the origin of the n-space. Equation 2.1 outlines how to create a proper Input Data Matrix.

$$IDM = \begin{bmatrix} Image_1\, pixel\, 1 & \cdots & Image_1\, pixel\, n \\ \vdots & \ddots & \vdots \\ Image_m\, pixel\, 1 & \cdots & Image_m\, pixel\, n \end{bmatrix} - \overline{IDM}$$

Equation 2.1. Forming the Input Data Matrix (IDM)

The next step is calculating the covariance matrix. Mathematically, a covariance matrix is the result of multiplying the transpose of the input data matrix with the original data matrix. However, the covariance matrix produced in this case would be of size n x n. For images, n is the total number of pixels in the image (length multiplied by width). Considering even a modestly sized image (256 pixels in height and 256 pixels in width), the covariance matrix would balloon to a size of 65536 by 65536. A matrix of this size is inconvenient to work with due to its size; it may even exceed the allotted memory of a computer. Fortunately, a shortcut is available to obtain a matrix that is more manageable. Rather than using the input data matrix, the matrix's transpose can be used (denoted as a matrix named A in Equation 2.2). Then, using A, another matrix (named L in Equation 2.3) can be computed to be of size m by m, where m is the number of images used in the input data matrix. This size is more desirable as the number of images used is often a couple magnitudes lower than the number of pixels in each image.

$$Cov = IDM'IDM = A'E\ A$$

Equation 2.2. Computing the Covariance Matrix of the Input Data Matrix (IDM), where E = a diagonal matrix of the eigenvalues of the IDM and A = IDM'.

$$L = A'A$$

Equation 2.3. Computing a matrix that is more manageable than the covariance.

The matrix L is thus a type of covariance that is more manageable to work with because of its smaller size (m by m instead of n by n). Finding the eigenvalues (denoted as v in Equation 2.4) of this simplified covariance matrix will result in eigenfaces, which is a matrix also of size m by m. When the eigenfaces are multiplied with A (the transposed input data matrix), all of the modes of variation become apparent (matrix U in Equation 2.5). Each column of U is a collection of pixels that describe common features of all the images in the input data matrix. The rightmost column of U is an image that contains the most common features amongst the input data while the leftmost column contains the least significant features of each image (usually random image noise). Due to this fact, a further reduction of data size can be achieved by flipping U horizontally (so the most significant variation mode is first rather than last) and by considering the first M columns. The optimal value of M can be determined through experimentation. The resulting matrix will be denoted as R in Equation 2.6, and contains the first M modes of variation for the input data matrix rather than using all of the modes.

$$Lv = vE$$

Equation 2.4. Finding the eigenvectors (v) for the simplified covariance matrix (L), where E are the eigenvalues of L.

$$U = Av$$

Equation 2.5. Finding all of the modes of variation (U) for the IDM.

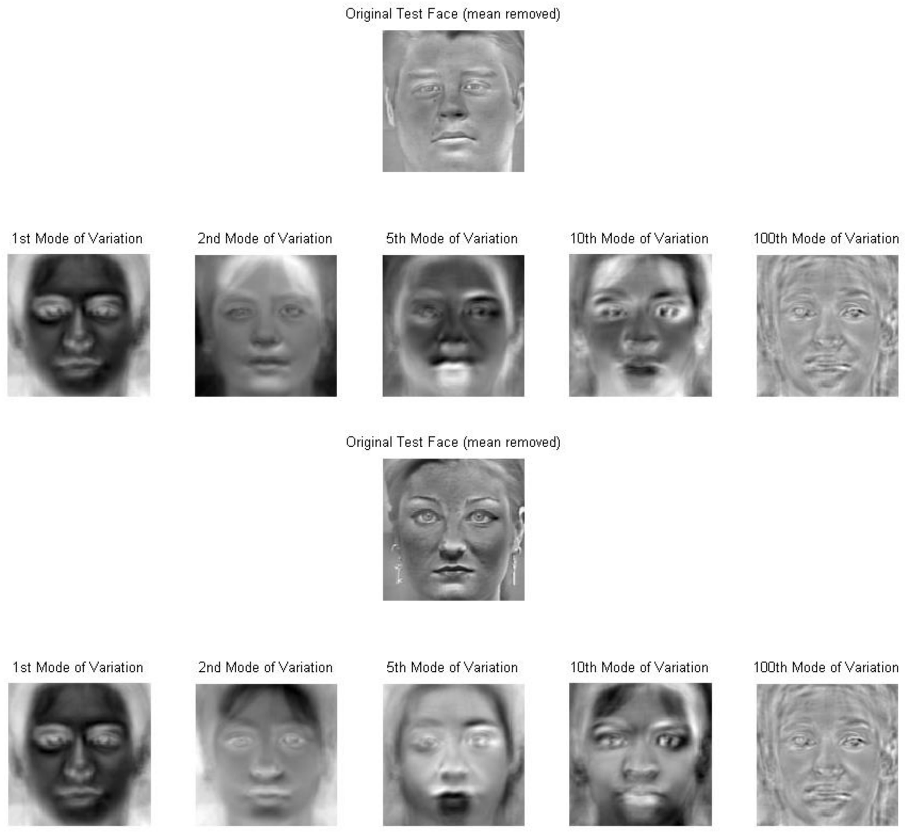


Figure 2.1. The modes of variation (eigenfaces) of the Cohn-Kanade dataset.

One more step remains in the process of retrieving features of a facial image using PCA. If the transpose of R is multiplied by a column vector containing the pixel values of a facial image, a vector containing M features is generated. The values in this vector are essentially the scalar values each mode needs to be multiplied by in order to reconstruct the input image. Reconstruction is possible because each mode of variation contains pixel values that, in some way, describe commonalities in the input data matrix. In fact, reconstructions that are very aesthetically close to the original image are possible if the image used was from the original input data matrix (as seen in Figure 2.2), because the modes of variation are ‘familiar’ with the image and its data is already imprinted in the eigenfaces to some capacity.

$$Features = R'A$$

Equation 2.6. Extracting facial features from the IDM for each image, where R is the first M columns of U.

$$Reconstruction = (R')(Image\ Features)$$

Equation 2.7. Reconstructing an image using the extracted PCA features.

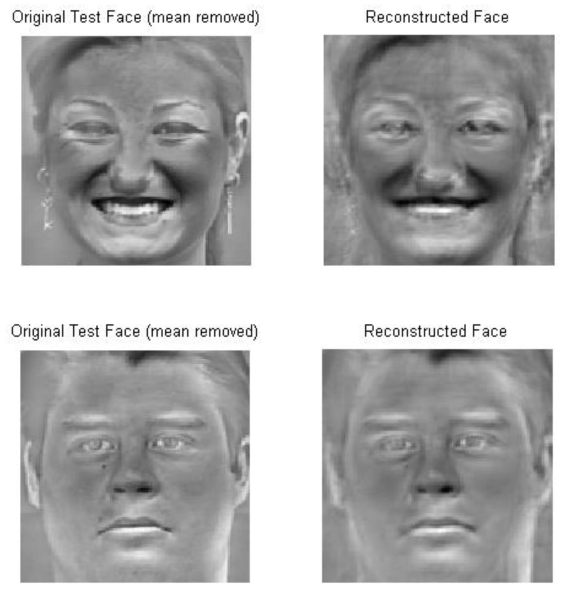

Original Test Face (mean removed)
Reconstructed Face
Original Test Face (mean removed)
Reconstructed Face

Figure 2.2. A reconstruction of a test image using images from a training set.

By extension, the modes of variation can also be applied to images that were not in the IDM. Depending on how representative the original IDM was and what the new input image is, the modes of variation do their best to describe the new image. If the new image is something that the was not accounted for in the IDM (like an image of a car when the

IDM was pictures of faces) the modes of variation will still describe the image based on what they learned from the IDM, but the reconstruction of the features will produce an image that looks mostly like a face but with a hint of the car.

In the case of emotion recognition, the IDM should contain faces with all ranges of expressions. This can be considered the training set, and the testing set should be comprised of facial emotions displaying images that are not in the training set. Using the steps described in this section, the features of each training and testing image can be extracted as a single vector of length M. Each of the training vectors should also have a label associated with them, which denotes the emotion that the features are supposed to represent. At this classification step, the aim is to give the testing vectors a label based on what was learned during the training stage. The classification step offers a wide range of options in terms of algorithms, and these different approaches account for the diversity in papers found during the literature survey.

The simplest method of classifying testing images using PCA features is by implementing a distance criterion. Imagining that the M-length vector of the test image exists as a point in M-space, and all of the training image vectors are points as well in the same space, the training point that is the closest to the test point (using Euclidean distance, for instance) can be assumed to share a label with the test point (Paithane, Hullyalkar, Behera, Sonakul, & Manmode, 2014). The proximity of the two points is what drives the assumption that they are from the same class. This method can be extended to look at the k-nearest neighbors of the test point, as well.

In fact, k-Nearest Neighbor (kNN) is another valid classification method that is noted in the literature (Busso, et al., 2004). Rather than taking the first nearest training point to the test point, the closest k points can be taken. The labels of these k points are then examined and the most common label (the arithmetic mode) is used as the class label for the test point. Due to this majority voting, an odd value for k is recommended.

Linear Discriminant Analysis (LDA) was also applied to PCA features for classification. However, one paper in particular included an extra step in the pre-processing phase for the input images that improved their recognition accuracy. The extra step addresses a problem that PCA has with diverse IDMs. To lower the variance in facial proportions amongst the training samples, each face was morphed to have the same mean shape. Even though the faces were showing emotion, factors that did not contribute to the expression (like face outline shape, nose location, and eye width) were slightly distorted in a way that created a consistent face shape (Calder, Burton, Miller, Young, & Akamatsu, 2001).

Discrete Cosine Transform. The Discrete Cosine Transform (DCT) is a frequency transform (similar to the Fourier Transform) that also compacts a signal (Bhadu, Tokas, & Kumar, 2012). In fact, it is commonly used for JPEG image compression. However, the DCT also has applications in image feature extraction, as it is referenced in many papers regarding emotion recognition using facial images. It can be applied in any number of dimensions, but the focus here will be on the 2D DCT.

$$X_{k1,k2} = \sum_{n1=0}^{N_1-1} \sum_{n2=0}^{N_2-1} x_{n1,n2} \cos\left[\frac{\pi}{N_1}\left(n_1 + \frac{1}{2}\right) k_1\right] \cos\left[\frac{\pi}{N_2}\left(n_2 + \frac{1}{2}\right) k_2\right]$$

Equation 2.8. Formula to calculate DCT on a 2D matrix.

In the equation above, Equation 2.8, there are several variables that need to be explained. $X_{k1,k2}$ is the resulting DCT coefficient, with k1 and k2 being the coefficient's position in the resulting 2D DCT matrix. N_1 and N_2 are the length and width (although the order does not matter) of the original image. Lastly, $x_{n1,n2}$ is the pixel value of the image at position n_1 and n_2. The same result $X_{k1,k2}$ can also be obtained by first taking the 1D DCT of each row and then taking the 1D DCT of each column of the first matrix.

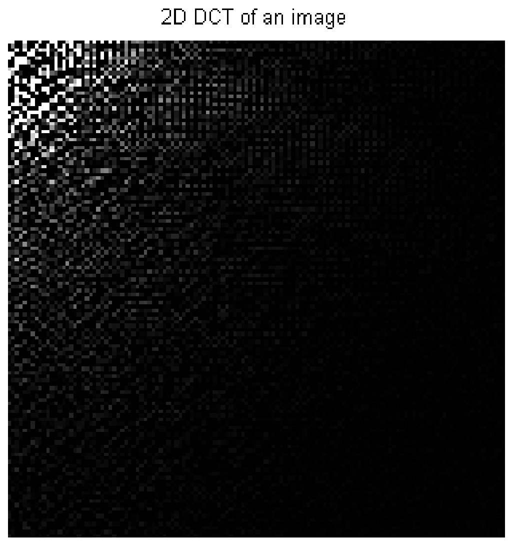


Figure 2.3. A sample transformation matrix after a 2D DCT was applied to an image. The intensity of the pixel is proportional to the corresponding matrix cell value.

Since the DCT is a signal compaction tool, it would be expected to find the most important information at the beginning of the coefficients rather than the end. In fact, many of the trailing coefficients can be truncated and the reconstructed signal (using an inverse

DCT) from the remaining coefficients would be unaltered, up to a point (Guney). This is how the compaction happens: by eliminating the latter coefficients (the low frequency information) of a DCT transformed signal. In Figure 2.3, most of the information is in the upper left corner of the matrix. As a result, most of the matrix can be ignored as long as the coefficients in the upper left are preserved.

In applications for this research, a 2D DCT matrix was not an appropriate form to express the features of an image. Rather, the DCT coefficients needed to be in a 1D vector to be usable by classification algorithms. To convert the 2D DCT coefficients into 1D DCT coefficients, a zig-zag scanning method was used. A visual representation of how this method collects coefficients can be seen in Figure 2.4. The rules for a zig-zag scan are such: Start at the upper left corner cell of the matrix, if the current coefficient is at the top row/last column, the next coefficient should be the coefficient to the right of it/below it and the scan should continue diagonally down. If the current coefficient is in the first column/last row the next coefficient should the coefficient directly below it/to the left and the scan should continue diagonally up. This process is repeated until the last coefficient at the bottom right is reached. Along the way, each coefficient is recorded in a new column in the 1D DCT vector. Since the upper left coefficients are scanned first, the leading coefficients of the newly formed vector will have the important high frequency content.

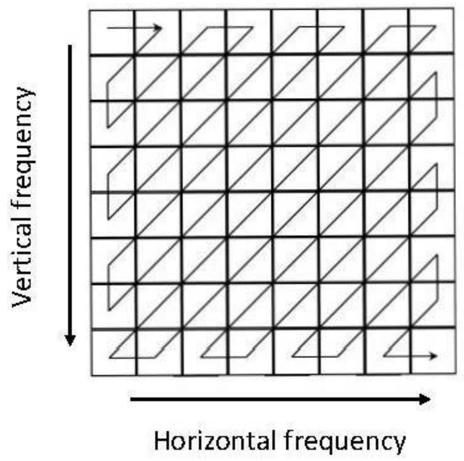


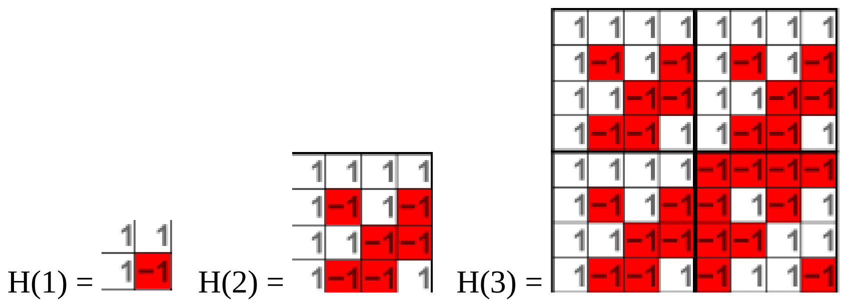
Figure 2.4. The process of collecting 2D DCT coefficients to form a 1D DCT feature vector.

Fast Walsh-Hadamard Transform. The Walsh-Hadamard Transform is another frequency-based feature extraction method used in this research. It is in the same vein of Fourier transforms although it uses a different basis function. Rather than the sinusoids previously seen in the DCT, WHT utilize square waves with peaks/troughs of ±1 (Hassan, Osman, & Yahia, 2007). The transform had shown up in literature ((Hassan, Osman, & Yahia, 2007) and (Faundez-Zanuy & Fabregas, 2007)) that discussed using it as a feature extraction method for facial recognition systems. This prompted the exploration of the WHT in the facial emotion recognition domain.

The key component of the WHT is the Walsh/Hadamard Matrix. It is a self-repeating sequence constructed by Equation 2.9 and contains only 1's and -1's. The most important property of the matrix is that its rows are orthogonal to each other. Since it only contains two simple numbers, the Walsh/Hadamard matrix can be easily generated and applied without using too much memory. The benefits of using the Walsh/Hadamard matrix come from its lowered computational complexity and speed (Faundez-Zanuy & Fabregas, 2007).

$$H_m = \begin{pmatrix} H_{m-1} & H_{m-1} \\ H_{m-1} & -H_{m-1} \end{pmatrix}$$

Equation 2.9. Formula for producing a Walsh/Hadamard Matrix (Hadamard Transform, 2016).



$$H(1) = \begin{matrix} 1 & 1 \\ 1 & -1 \end{matrix} \quad H(2) = \begin{matrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{matrix} \quad H(3) = \begin{matrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{matrix}$$

Figure 2.5. Walsh Matrices of different sizes (Hadamard Transform, 2016).

The input image must be of size 2^m by 2^m in order to perform the transform (it can be zero padded to meet this requirement). The *m* is used to determine how many recursions are required to generate the Walsh/Hadamard matrix. To get the transformed image, Equation 2.10 is used (Faundez-Zanuy & Fabregas, 2007). The transform is orthogonal, symmetric, involutive, and linear (Hadamard Transform, 2016) and produces a result that is the same size as the original image. The product is also zig-zag scanned because of the way that data is arranged in the output matrix.

$$T = H_m U H_m \text{ (where U = input image)}$$

Equation 2.10. Formula for performing the Walsh-Hadamard Transform (Faundez-Zanuy & Fabregas, 2007).

Facial Action Coding System. Paul Ekman's body of work can be referenced again when exploring the Facial Action Coding System (FACS). Through the use of Action Units (AUs), each possible facial expression is given a label. Since an emotion is a combination of multiple facial movements, FACS allows for emotions to be described by these universal measures (AUs). The underlying methodology for annotating the movements comes from physiology. Muscles under the skin of the face drive facial expression, and FACS accounts for all of the muscles on the face and their possible movements. The AUs specifically label these muscle positions. As a result, FACS is not a mathematical process as much as it is a holistic approach to represent emotions based on universal facial expressions (Ekman, Friesen, & Tomkins, Facial Affect Scoring Technique: A First Validity Study, 1971).

Forty-six of the AUs deal specifically with facial muscle movements. The AUs consider movements at sites like the inner and outer brows, the eyelids, cheeks, the nose, all parts of the lips, and the jaw. The entire range of motions for each body part is captured with Action Units. For instance, the lips are capable of pulled by the corners, funneled, depressed, tightened, puckered, and stretched. More importantly, these motions directly correspond to muscle activity under or near the lips like the Zygomaticus major, Labii superioris, and Orbicularis oris (Facial Action Coding System, 2016). This level of attention to detail is seen in throughout FACS, for all of the possible movements that facial features can make. On top of that, there are additional AUs outside of facial muscles which include the neck and the eyeballs.

The main focus of using Action Units is to fully describe how emotions are manifested on a face. Therefore, each facial image can be described as a composite of several AUs. In the case of the prototypical happy face AU 6 and 12 are most commonly observed. This correlates to a cheek raiser motion and lip corner puller respectively. The full list can be seen in Table 2.1.

Table 2.1

Emotions annotated using the Facial Action Coding System (Facial Action Coding System, 2016).

Emotion	Action Units
Happiness	6+12
Sadness	1+4+15
Surprise	1+2+5B+26
Fear	1+2+4+5+7+20+26
Anger	4+5+7+23
Disgust	9+15+16

FACS has several applications in varied fields of study. Most prominently, FACS aids with the analysis of emotion in psychology. It provides a standardized measurement and allows for accurate descriptions of facial gestures amongst researchers. FACS also shows up in animation disciplines. Specifically, 3D animation studios rely on FACS to generate convincing expressions for computer-generated faces (Villagrasa & Susin, 2009).

In the criminal justice system, FACS can give clues to a person’s honesty. The system is robust enough to describe microexpressions: the involuntary and hardly noticeable gestures on a face. Microexpressions often occur when the face wants to display an emotion but the person actively attempts to conceal it. Commonly, this practice occurs during lying, and criminologists who can detect and document them benefit from the knowledge (Microexpressions, 2016).

Chapter 3

Methods and Experimental Procedure

Basic Approach to Emotion Recognition

There is a general methodology that can be followed when performing emotion recognition on human faces. Such a framework is mirrored in many of the papers reviewed in the literature survey. The methodology can be seen as a series of steps that end with a system that classifies emotion based on facial expression. These steps may seem trivial, but they offer a wide array of possible experimentation, of which this thesis only touches on some of those combinations. The system flow can be seen as: choosing an appropriate facial image database, isolating the face from the rest of the image, preprocessing the cropped result, extracting features from that image, and classifying the features. This “Methods” chapter will examine these steps, providing both the background and application of each. However, before starting the methodology discussion, it is important to consider the three fundamental approaches to facial emotion recognition.

As previously stated, there are many available possibilities when choosing ways to extract features or classifying data relating to facial expressions. Those choices are best informed when taking into account the type of data that the emotion recognition system will work with. Specifically, the systems can take a geometric, appearance, or 3D based approach. Each has its own strengths and weaknesses, which will be discussed now.

A geometric approach to emotion recognition involves setting landmarks on a face and tessellating lines amongst them. The locations of the points and lengths of the lines are used as a model to represent the face. These models will vary from person to person given different facial structures, but an even greater variation will be seen given different expressions on those faces. Already, a weakness presents itself in the fact that person-to-person differences need to be filtered out in order to have more accurate classifications. Furthermore, if the face is partially obscured in any way, the geometry at the covered region may not reflect real life.

Instead of using points and lines, the actual appearance of the face can also be used for feature extraction in an emotion recognition system. In this second approach, the pixel values of the image are used. The pixels create a 2D matrix, and a vast range of transforms can be applied to the matrix to extract information. The image as a whole can be transformed (global feature), or it can be broken into parts (like mouth and eyes, creating local features). The effectiveness of this method is dependent on the implemented algorithm the downsides include possibly higher computational times than the previous geometry-based approach.

The previous two approaches consider 2D images as inputs, but a third option takes this one dimension further. There are several ways to catch a human face in three dimensions. Some methods include using cameras at various angles to create a composite 3D reconstruction (Zheng, 2014) while others create a 3D model by bouncing infrared light off of the face (like Microsoft's XBOX Kinect peripheral) (Wang, et al., 2010). Regardless of the capture process, the result is a 3D wireframe representation of a face. The downsides are already present: higher capture complexity and more computational intensity due to higher volume of data points. However, the benefits include important details that are sometimes hidden in 2D images, like brow or cheekbone location.

Facial Image Database

Selecting a facial image database is an important step in the creation of an emotion recognition system. The factor to consider when choosing a database are application. Application is the answer to the question: what will the emotion recognition be used for? In cases like security system, lower resolution images are more permissible to use because it would be expected that the security cameras would capture faces that are farther away. If the setting is outdoors, training sets should include faces that are lit in a multitude of different ways and from different angles. In applications like medicine and marketing, databases that feature high resolution images with even lighting are appropriate. In terms of approach, if the system will utilize 3D models, then a respective database needs to be implemented.

Regardless of the application, a database needs to meet several criteria to be considered a useful training set. Since the topic of discussion is emotion, the images need to show a variety of expressions. They should reflect how people emote in natural circumstances. In cases where the emotion images are derived from organic sources like surveillance footage or photo-albums, they are known as ‘unposed’ expressions. Otherwise, actors are used to create ‘posed’ expressions by re-enacting them in front of a camera. Ideally, ‘unposed’ expression databases hold more value because they are better representations of real life. On the other hand, posed expressions are much easier to attain and can be near-genuine if the actors are skilled enough.

Databases should also offer discrete labels for their training images. These labels are often in the form of the basic emotion being portrayed by the subject (like neutral, happy, sad, or angry). This allows for straightforward class creation because each label can be a distinct discrete number. Outside of images and labels, anything else that a database can offer is an added bonus. Some databases offer annotated landmark positions. This means that each picture contains x and y coordinates of points placed in the same spots across all faces (like around the eyes, nose, and mouth). Other databases could also contain FACS information for each face. These are items that could make emotion classification more versatile if implemented.

Cohn-Kanade Database. For this thesis, a 2D facial image database was used. Since the application was research-based, such a database provided an undistorted setting to test different algorithms. The name of the chosen database is the Extended Cohn-Kanade. It was compiled between the Robotic Institute at Carnegie Mellon University and the Department of Psychology at the University of Pittsburgh. The database features facial images from both men and women (123 subjects) with a diversity split of 81% Euro-American, 13% Afro-American, and 6% other. In total, 593 sequences were recorded across all of the subjects. In the context of the database, a 'sequence' refers to a set of images where the subject started with a neutral facial pose and ended with the requested emotion. The sequences were short video recordings that were decomposed into individual frames for the database. The emotions portrayed in the dataset are: Neutral, Angry, Contempt, Disgusted, Fearful, Happy, Sad, and Surprised (Lucey, Cohn, Kanade, Saragih, & Ambadar, 2010).

As for the individual images, they came in sizes of either 640 by 480 or 640 by 490 pixels. The majority were in a grayscale while a select few were in color. Each image came with a set of landmarks: 68 points placed on specific locations on the face, which move with those locations as the face morphs between emotions. These points were placed automatically on the face using Active Appearance Modeling (AAM). The Extended Cohn-Kanade database also featured labels for each sequence, which were visually inspected by humans to make sure the performed emotion was the same as the requested one. Lastly, each end image in a sequence was coded using FACS. To get the Action Units for the face, the AAM landmarks were used in conjunction with a Support Vector Machine (SVM) classifier (Lucey, Cohn, Kanade, Saragih, & Ambadar, 2010).

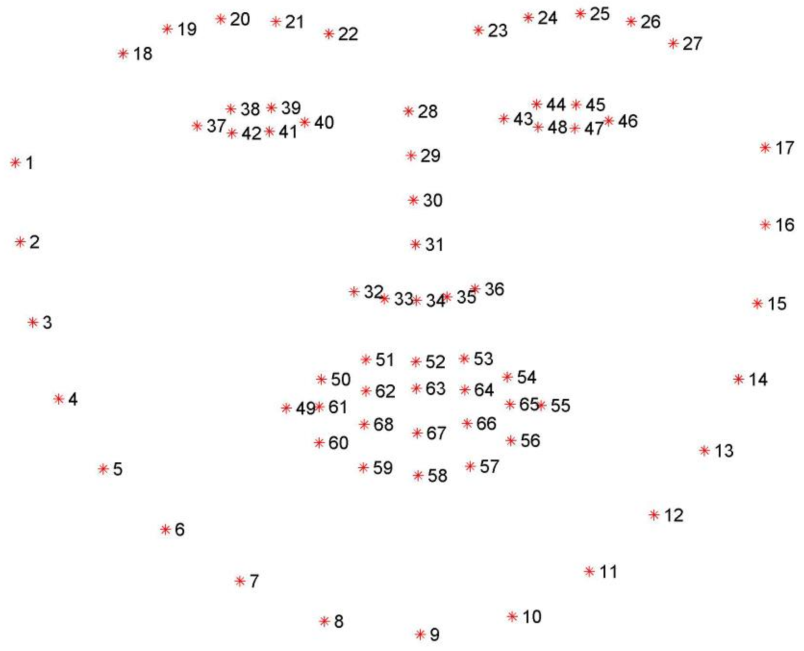


Figure 3.1. 68 landmark points used in Extended Cohn-Kanade database (Lucey, Cohn, Kanade, Saragih, & Ambadar, 2010).

In the context of this work, the Extended Cohn-Kanade database was used with several considerations. First, the Contempt emotion was not used due to the small sample size. This resulted in the remaining seven emotions being used: Neutral, Angry, Disgusted, Fearful, Happy, Sad, and Surprised. To get samples for these emotion, the first and last images from each sequence were taken. Since each sequence starts with a neutral pose and ends with a fully posed emotion, the number of neutral samples was the size of all the other emotions combined. This disparity in sample size is addressed in a future section, 3.4 Synthetic Minority Oversampling Technique. Lastly, errors in labeling were found in some sequences after a manual inspection. For those sequences, the labels were updated to more accurately reflect the emotion being displayed. The below table, Table 3.1, shows the final sample sizes for each emotion after all of these considerations were made.

Table 3.1

Breakdown in sample size for each emotion in Cohn-Kanade database.

index	emotion	sample size
0	netural	526
1	anger	41
2	contempt	0
3	disgust	41
4	fear	19
5	happiness	61
6	sadness	22
7	surprise	75
8	unclassified	267

JAFFE Database. The second facial expression database used in this research is the Japanese Female Facial Expression (JAFFE) dataset. This database was created at Ritsumeikan University in 1997 and features 10 subjects who are, as the name implies, Japanese females. Emotions portrayed in this database are: Neutral, Angry, Disgusted, Fearful, Happy, Sad, and Surprised and there are around 1-3 samples of each emotion per subject. There is a total of 213 images, sized at 256 by 256 pixels. Unlike the Cohn-Kanade database, there was no FACS coding for the faces and no landmark points were present.

The JAFFE database was used to test the robustness of implemented algorithms in terms of cultural differences. It was apparent that some of the ways that the Japanese portrayed emotion was different compared to Americans. Namely, puffed cheeks were much more common when displaying anger for the former. JAFFE also provided a blank slate to build the automatic facial point annotator, since there were no annotated landmark points for the faces. This automated point placement utility was tested on this database, which is explained in future section, Section 3.7 (Automated Point Placement of Facial Landmarks).

Table 3.2

Breakdown in sample size for each emotion in JAFFE database.

index	emotion	sample size
0	netural	30
1	anger	30
3	disgust	29
4	fear	32
5	happiness	31
6	sadness	31
7	surprise	30

Facial Detection

After choosing the right database, the next stage in an emotion recognition system is facial detection. Since it is not always the case that the facial images will be framed perfectly in such a way that only the face is present with no background, there needs to be a facial detection step. This was the situation with both the Cohn-Kanade and JAFFE databse. The goal is to a have a facial image that is free of extraneous information. If this is not the case, unwanted features in the image’s background may be extracted and classified along with important information, resulting in error. A widely used algorithm that can extract a face out of a scene is the Viola-Jones algorithm. This algorithm was further chosen for the research because of its readily available implementation in MATLAB.

The Viola-Jones algorithm is a robust process that can detect any object that it is trained on, not just faces. Furthermore, it performs the detection in real-time. It was first introduced in 2001 by Paul Viola and Michael Jones in a paper submitted to the Second International Workshop on Statistical and Computational Theories of Vision – Modeling, Learning, Computing, and Sampling. In this paper, a 4-step approach to object detection was explained. The steps are Haar feature selection, integral image creation, Adaboost training, and cascaded classifiers.

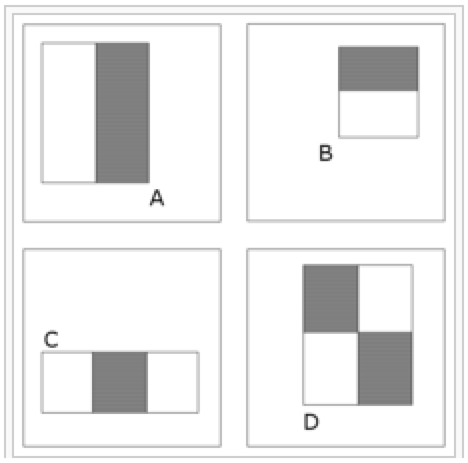


Figure 3.2. Feature types used by Viola-Jones Algorithm (Viola-Jones object detection framework, 2015).

In the above image, these rectangular patterns are placed on various locations on the image, and the sum of the pixel values in the light regions are subtracted from the pixels in the dark regions. Feature types A and B are for vertical and horizontal features, respectively, while type C can be used to represent areas like the nose and type D for diagonal features. A mathematical trick in the Integral Image stage allows for the computation of the features to be hastened. Since the feature types are only of size 24 by 24 pixels, a high magnitude of computation would have to take place to cover the entire image. However, Adaptive Boosting (Adaboost) learns from rejected regions and concentrates on those that give better results (Viola & Jones, 2004).

Lastly, the cascade structure puts several of these processes in series. The stages can separately have lower detection accuracies because their serialization contributes to an overall increase in the final detection accuracy. Since a face will tend to make up a small portion of the image, these stages throw out most of the information and only focus on true and false positives. Each additional stage chisels away more false detections until only mostly accurate data is left (Viola & Jones, 2004).

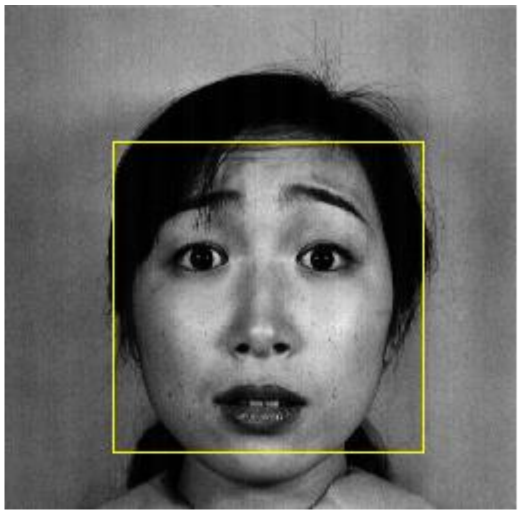


Figure 3.3. Viola-Jones algorithm cropping out faces in an image.

The result is a region in the image that most likely contains a face. This is informed by the area where the feature types closely resemble trained data. The figure above shows the usual output: a bounding box surrounding the region of interest ready for cropping.

Synthetic Minority Oversampling Technique (SMOTE)

An important aspect of training a machine learning system is to ensure that each class is well represented. In the case of this research, the classes are the different emotions to be recognized. As previously mentioned, the samples gathered from the Extended Cohn-Kanade database showed a disparity in sample size amongst the emotions. In some cases, the neutral sample size was an entire magnitude larger. One way to deal with unevenness in datasets is to under-sample the majority data. However, Synthetic Minority

Oversampling Technique (SMOTE) addresses the problem by inflating minority data to match sample sizes with majority data (Chawla, Bowyer, Hall, & Kegelmeyer, 2002).

The primary purpose of SMOTE is to create new samples for a dataset using data that is already in that dataset. There is a wide range of ways to accomplish this without SMOTE, like taking real data and rotating or scaling it slightly to create new samples (Bunke & Ha, 1997). While these samples may be 'synthetic' they are still representative of normal conditions for the class. SMOTE, on the other hand, is a method that involves artificially creating data from existing samples.

The primary inputs for the SMOTE algorithm are as follows: the minority data as a 2D matrix where each sample is a row vector, a value of k for the k nearest neighbor that is used, and an integer N by which the size of the data will be inflated by. If this value of N is 1, the sample size for the minority data will stay the same. If N is 2, the sample size will double, and so on. The value of N can also be between 0 and 1, but that will result in an under-sampling of the data. The output of SMOTE will produce a new 2D matrix that features the old minority data along with new synthetic samples.

The process takes one original minority sample at a time. It evaluates that sample's distance (typically a Euclidean distance metric) to all other original minority points and takes the k nearest neighbors. With this cluster of samples, it takes the central sample and a randomly chosen neighbor to create a synthetic point between them. Specifically, the difference between the two points is multiplied by a random number between 0 and 1 and this value is added to the central sample. The new vector is a synthetic sample.

$$synthetic = center + rand(0,1) * (center - near\ neighbor)$$

Equation 3.1. Formula for creating a synthetic point using two other points (Chawla, Bowyer, Hall, & Kegelmeyer, 2002).

Equation 3.1 is the technical representation of how a synthetic point is generated. Another way to consider how the new sample is made is to think of a straight line that stretches between the central point and its k^{th} nearest neighbor. On that line is an infinite amount of possibilities, and SMOTE randomly chooses one of them to create the synthetic sample. As a result, the new sample is still representative of the original data and does not create new outliers since it is contained between two points. The process is repeated until there are no more nearest neighbors for a central point at which time a new central point is chosen. That new point is still part of the original minority data and its k^{th} nearest neighbors are used to create more synthetic data. The process is over until enough synthetic data has been generated to meet the size established by N in the input (Chawla, Bowyer, Hall, & Kegelmeyer, 2002).

Preprocessing

The pre-processing step is one that enhances data by making various adjustments to it. The aim of pre-processing is to minimize variability of all aspects except those that relate to emotion being displayed on the face. Such aspects could be lighting differences, skin tone, image resolution, or face size. For this research, preprocessing was performed on images of faces that were already cropped using the Viola-Jones algorithm. In the preceding section it is assumed that the input is strictly a face with minimal background.

Scaling. Depending on the camera placement and size of the subject's face, the resulting cropped image from Viola-Jones will not be a consistent image size. In this pre-processing operation, each newly cropped image is scaled to be a standard size. While scaling is not a major issue for frequency transforms like the DCT, it was important to have the images be the same size for Principle Component Analysis. To generate the Input Data Matrix, each row needs to be the same length or else the IDM cannot be generated. If the images are not all uniform in size, then each row will have a different length. Several standard image sizes were tested: 256 by 256, 128 by 128, and 64 by 64. Since the pre-processed training images were used for both systems (DCT and PCA), the scalings were implemented in both to see the effects. Further discussion of the results can be found in the next chapter.

Patching. During the course of this research, local features were investigated to see if they aided in system accuracy. These local features were manifested through the use of patching. It was necessary to have automatically placed localized facial points for this step. Patching involved sampling the picture at these points using a predefined window. Specifically, each facial landmark was used as a center for a square that cropped itself out of the original image. As a result, the original facial image could be broken up into 'frames' and the frames could be analyzed separately. Examples of what these frames/patches looked like can be seen in the preceding figure, Figure 3.4.

Figure 3.4. Examples for facial patches.

Each patch was a size of 35 by 35 pixels. Due to the uniformity of image size ensured by the previous scaling step (Section 3.5.1), the amount of data captured by each patch was roughly the same. This means that a patch centered at an eyebrow corner contained roughly the same amount of eyebrow across all samples. Aside from eyebrow corners, the following were also used for patching: lip corners, eye corners, and lip centers. The philosophy behind using these locations goes back to lessons learned in the literature survey. These particular locations on the face show the most variability between emotions. As a result, concentrating on just those regions would allow for a more detailed description of how the emotion is displayed on the face. In the subsequent experiments that utilized them, the patches were treated as separate images and they were given labels that were the same as the parent image that they were taken from.

Normalization. Whether the input at this pre-processing step was a full face or a patch, normalization was performed to tweak the brightness of the image. The reason behind changing the brightness was to compensate for differences in lighting conditions and skin tones amongst images. A variety of skin tones were represented in the Extended Cohn-Kanade database which increased the variability of the data to an unwanted degree. To correct the brightness of each image, a constant value, *c*, was either added to or subtracted from the original. This constant *c* was calculated by first taking the mean of the pixels in the image twice to get a scalar: the overall mean brightness. Since the images were 8-bit grayscale (ranging from values of 0 to 255), the desired mean brightness should be 127. Thus, the constant *c* was obtained after the mean brightness was subtracted by 127. If the image was too bright, *c* would be positive number, and each pixel value would be subtracted by that amount. Conversely, dark images were brightened after *c* was added to each pixel. The result was a 2D matrix with mean brightness of 127 for each image, effectively minimizing the influences of lighting and skin tone. Figure 3.5 below demonstrates the visual changes in the image when it was normalized.

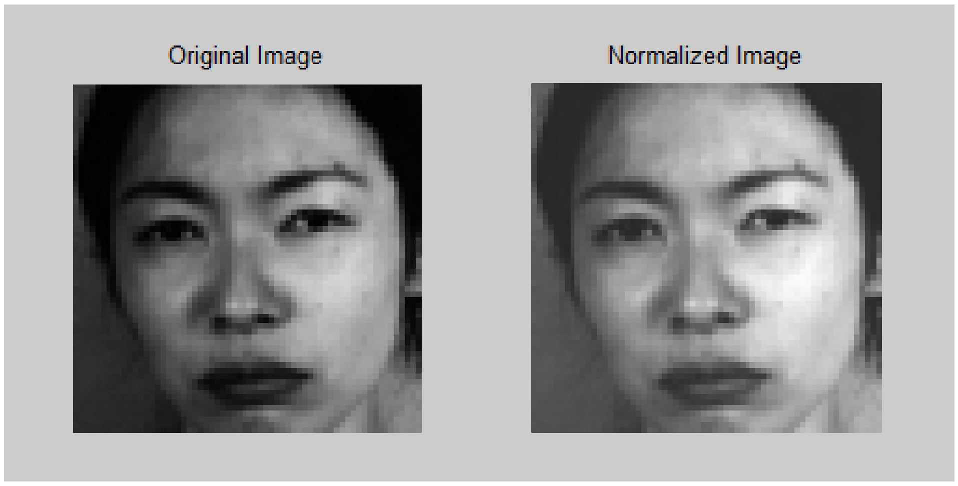


Figure 3.5. Effect of normalization on an image.

Energy Thresholding. The last pre-processing step that an image went through in the emotion recognition system was the Energy Thresholding stage. The purpose of this step is to ensure only images that contain ‘information’ are used in training and testing. In the context of this research, ‘information’ is the energy of the image, where higher energy images are more visually complex. In Figure 3.6, the difference between a low energy and high energy can be seen. This step was only applied to facial patches because there was the possibility that some patches did not catch any facial features, resulting in a relatively blank image. Blank images were not valuable to the system because 1) they did not contain features for the feature extraction phase and 2) their classification could have fallen into any category that also had blank images trained in it. Energy thresholding ensured that all samples were actually usable.

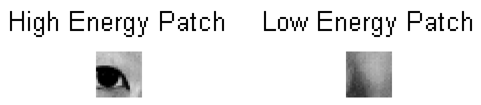


Figure 3.6. High energy vs. low energy image.

To find the energy of a patch, the patch image was transformed into the frequency domain and the sum of the squared frequencies was taken. This procedure is reflected in Equation 3.2. In it, X(f) is the image after it has been transformed (using a DCT in this case) and turned into a 1D vector with length N. It was visually confirmed that images that were visually complex had a higher energy compared to bland images, so this calculation was valid. Lastly, a threshold had to be set to exclude low energy images. All possible patches from the Extended Cohn-Kanade were taken and their energies were calculated. A value was picked that would throw out up to 10% of all the patches, ensuring that a stable majority was still usable. A visual inspection of the denied patches confirmed that no visually complex samples were needlessly discarded.

$$E = \sum_{f=0}^{N-1} X(f)^2$$

Equation 3.2. Calculating the spectral energy density of a signal.

Feature Extraction

At this stage of the system, the input images have been pre-processed and are ready to have their features extracted. The goal of feature extraction is to represent each image as a single vector, where each dimension retains some important information about the original image. A useful method of feature extraction should ideally be able to reduce commonalities and noise in a dataset while emphasizing what makes the input unique. The facial images being used in these experiments can be viewed as two dimensional arrays of pixels, and feature extraction algorithms need to condense that information down to a single vector. This can be achieved in multiple ways, some of which will be detailed in the following sections.

DCT/DST. The Discrete Cosine Transform (DCT) and Discrete Sine Transform (DST) were both used as feature extraction methods for this research. Either transform required just an image as input and both produced a 1D feature vector as output. It was previously discussed in the DCT section of the literature survey that these transforms move the image into the frequency domain and contain important high frequency information at the start of the vector. After a zig-zag scan is performed to convert the resultant 2D coefficient matrix into a 1D vector, experiments had to be carried out to determine the optimal length of the feature vector.

Since the latter coefficients of a DCT/DST feature vector contain low frequency information, it is possible to truncate them without too many negative consequences on system accuracy. A determination had to be made, however, of how many of these low frequency coefficients could be thrown out before accuracy was impacted. To do this, images were transformed and only the first 5 coefficients were kept. The system was trained and tested with feature vectors of length 5 and the final accuracy was produced and recorded. Then, this procedure was repeated but with features vectors of length 10. Each time the procedure was repeated, the length of the feature vectors was raised by 5 until the length was 150. The accuracy from each experiment was recorded and a plot was generated that graphed accuracy versus feature vector length. The optimal coefficient length was chosen by locating the point on the graph where the coefficient length was the lowest before the accuracy began to plateau or decrease. Figures in this thesis like Figure 4.1 will be common and were used to find an optimal coefficient length.

The DST is a companion function to the DCT in the sense that it uses sines instead of cosines. Otherwise, they are set up identically as Equation 3.3 shows. The resulting transformations are similar, but with some differences. There was no literature that mentioned using a DST so an attempt was made to compare it to the DCT in this research.

$$X_{k1,k2} = \sum_{n1=0}^{N_1-1} \sum_{n2=0}^{N_2-1} x_{n1,n2} \sin\left[\frac{\pi}{N_1}\left(n_1 + \frac{1}{2}\right)k_1\right] \sin\left[\frac{\pi}{N_2}\left(n_2 + \frac{1}{2}\right)k_2\right]$$

Equation 3.3. Formula to calculate DCT on a 2D matrix.

MATLAB was used to implement all aspects of this feature extraction algorithm. The software provided a function that performed a 2D DCT to the specifications of this system. This meant that the function accepted an image (2D matrix containing pixel values) as an input and output a matrix of the same size which was the final transformation. Initially, only full faces were used as input. However, after facial patches were introduced into the system, they too were used as inputs for the frequency transformations. Although the two types of images differed in size, they still produced output vector of the same length due to zig-zag scanning.

The zig-zag algorithm had to be coded from scratch but followed the simple rules that were previously explained in Section 2.2.2. As an input argument, the desired vector length was used along with the 2D DCT product and the output was a 1D vector version of the input. This simplicity in implementation was a big motivator for testing the DCT.

Principle Component Analysis. Due to the prominence of Principle Component Analysis and Eigenfaces in both facial and emotional recognition literature, an attempt was made to implement it in this system and record the results. As previously stated in Section 2.2.1 the goal of PCA is to find common modes of variation amongst a set of data and express each piece of data as a combination of those modes. The result is a dimensionally reduced feature vector that can be classified by comparing its modes with others.

MATLAB's PCA function was not used, but rather the steps outlined in Section 2.2.1 were used to perform the analysis. Each image was turned into 1D vector by scanning the rows in order. After all of the training images were shaped to that form, they were concatenated vertically into one matrix, the IDM (Input Data Matrix). In this IDM, each row was a 1D version of a training image. The IDM then had a number of transformations that could be done to it to find different values, the most important of which was the matrix that contained all of the modes of variation.

Experiments were carried out to reduce the number of columns of the modes to further reduce the dimensionality. Similar to the DCT experiments, several columns were added at a time and the accuracy was assessed. When the accuracy plateaued or decreased, the lowest number of columns at the start of the plateau was used as the final number of modes/length of the feature vectors. Both LDA and kNN (with a k value of 1) were used to classify the feature vectors.

The feature vectors for all training images were instantly computed when the modes matrix was multiplied by the IDM. Each row in the result was a feature vector for each row that was in the IDM. The feature vectors were separated and each row went into the classifiers one by one to generate the respective models. To create feature vectors for the test data, each test image was also shaped into a 1D vector and that vector was multiplied with the same modes matrix. There was no separate modes matrix for the test data because the assumption was that the test images were visually similar enough to the training images that the modes for the training would be enough to express the testing as well. The result of the test image and mode matrix multiplication was a feature vector for the test image that could be used in a classifier to give the test image a label.

PCA was not used on facial patches. There were not enough distinguishing features in facial patches for PCA to identify distinct modes of variation. Only full facial images were used as inputs for this process.

X-Point Method. A novel approach was also attempted in this research to classify emotions from facial expressions. Using prior FACS knowledge, it was hypothesized that merely looking at distances between key points on the face could simplify the process of feature extraction. For instance, during a smile, the corners of the lips are lifted, and the distance between these corners and the corners of the eyes is minimized. On the other hand, with a frown, the lip corners are pulled down, making the distance from the lip corners to the eye corners larger. It was hypothesized that perhaps only considering such distances was enough to create a reasonably accurate emotion classification system.

As a starting point, 14 points were chosen that corresponded to important locations on the face and 7 lines were drawn amongst them. This can be seen as the first configuration in Figure 3.7. To neutralize the variability of different image scalings, these distances were all divided by the same number: the distance between the eyes. It was assumed that this particular distance would remain the same despite different emotions being displayed.

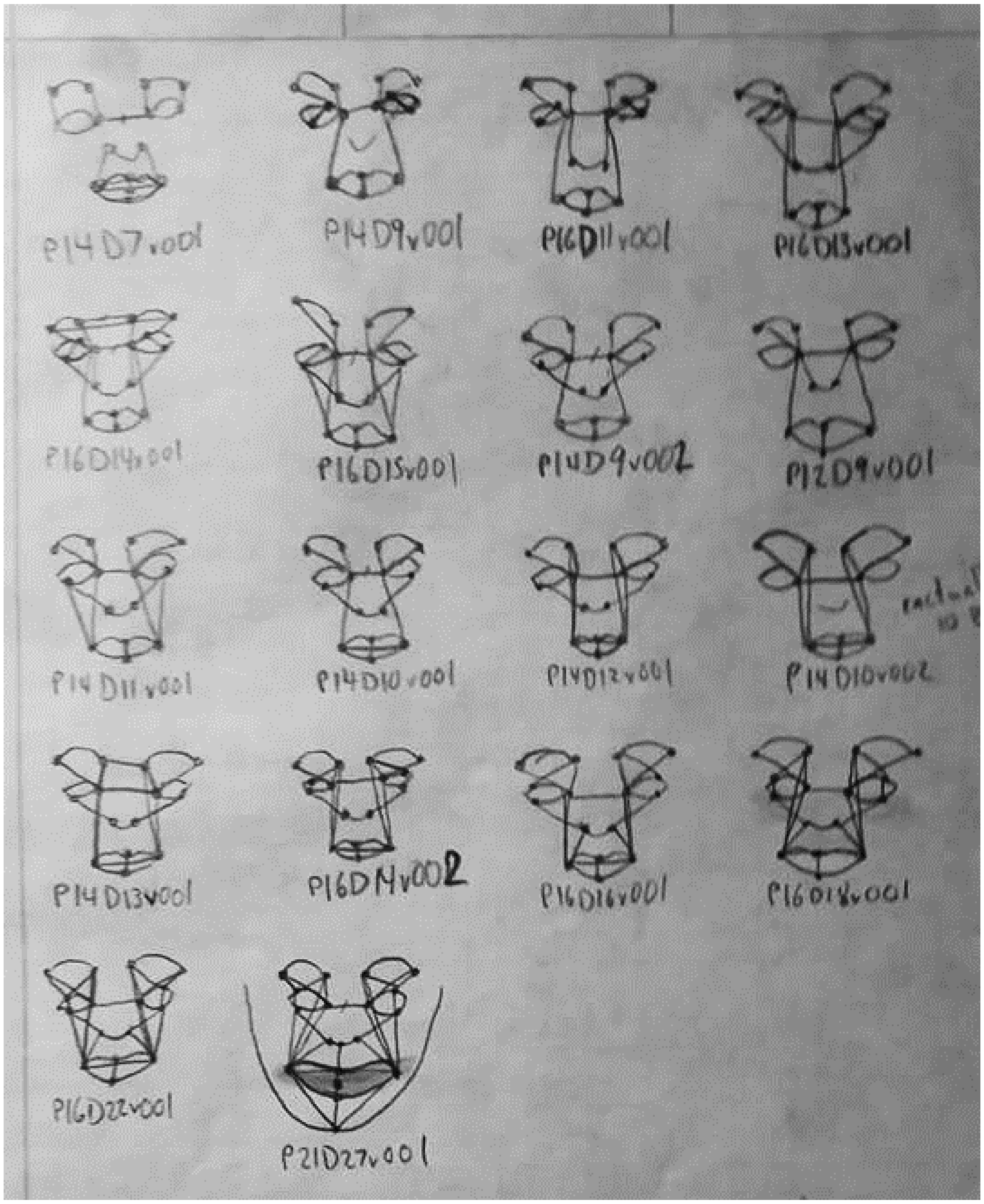


Figure 3.7. All attempted configurations of the X-Point method.

Initial experiments placed 14 points but only used 7 lines drawn amongst them. They were chosen due to their variance between different emotions (such as how eyebrows rise or lower in respect to the eyes). However, there were many more possibilities for points and lines, so a process of gradually adding more points and lines took place. Each new point/line setup was a trial and error experiment. In some cases, the same number of points was kept while extra lines were drawn amongst them. In other experiments, new points were added and new lines were drawn for those. The most obvious lines were drawn first: those that instinctively felt as if they would change the most between expressions. Based on the results after each experiment, lines were either added or removed to see their impact on system accuracy.

By the end of the experiments, a local optimal configuration of points and lines was found. Since not all points and lines were attempted (as time would not permit), the optimal configuration can only be designated as a 'local maximum' for the accuracy vs. dimensionality curve. As with most other feature sets, both kNN and LDA were used for classification and compared.

Automated Point Placement on Facial Landmarks

Over the course of the research, the need arose to have points automatically placed on key areas of the face. As previously stated, the Cohn-Kanade dataset provided 68 points as supplemental information for each image. The JAFFE dataset did not contain any landmark points, so a tool had to be implemented for automatic annotation of the facial images.

The methodology behind the Cohn-Kanade database's automatically placed points was examined to gain an understanding of how to reproduce it for this research.

Documentation in the database noted that Active Appearance Model (AAM) was implemented. Further research on the topic revealed that AAM was a more sophisticated version of ASM, Active Shape Model. In both cases, Principle Component Analysis was used to extract the dynamics of the placed points across the entire set. The points were first placed manually on the faces in order to train the system to place them automatically in the future. ASM used the position data of the points to find the different modes of variation amongst them (Cootes, Taylor, Cooper, & Graham, 1995). AAM also used the position of the points, but as the name would imply, also found modes of variation in the appearance of the images to guide correct point placement (Cootes, Edwards, & Taylor, 1999).

For the sake of simplicity, the less complicated approach to automated point placement (ASM) was implemented for this system. There was no need to make the automated point placement overly accurate or computationally quick due to the scope of this work. As long as the points were close to where they needed to be, then ASM would be acceptable. The fact that ASM implemented PCA was another big benefit because PCA was already examined and understood as a method for feature extraction in this research. The overlap in methodology here minimized the time it took to implement an automatic facial landmark annotator. The rest of the discussion in this sub-section will revolve around how the annotator was constructed.

The bulk of the automatic point placing tool comes from a 2004 paper written by T.F. Cootes and C.J. Taylor entitled ‘Statistical Models of Appearance for Computer Vision.’ However, as it will be later discussed, there was some room for a novel algorithm to be inserted into the process. ASM functions with three main algorithms: a fitting tool, a molding tool, and a search tool. These processes take a cloud of points and gradually guide them to their appropriate positions on the face. For future reference in the context of this research, a point cloud is a set of 68 two dimensional points.

ASM Fitter. The primary focus of the fitter is to align one set of points to another set using translation, scaling, and rotation operations. The fitter can also be used to find the mean face in a set of many faces. Using Equation 3.7, the x and y coordinates of individual points are transformed by three parameters: the scale factor s, the rotation factor θ, and a translation factor. The translation factor can be computed by taking the mean ($\bar{x}$) of all x coordinates in a point cloud and mean ($\bar{y}$) of all y coordinates, and subtracting $\bar{x}$ from each x coordinate and $\bar{y}$ from each y coordinate. This operation centers all of the points in that point cloud about the origin.

$$\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} s\cos(\theta) & s\sin(\theta) \\ -s\sin(\theta) & s\cos(\theta) \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} + \begin{pmatrix} \overline{x_1} \\ \overline{y_1} \end{pmatrix}$$

Equation 3.7. Operation to move a 2D point x_1,y_1 to x_2,y_2 (Cootes, Taylor, Cooper, & Graham, 1995).

The scale factor, s, will uniformly shrink or expand the points about their center. Values of s between 0 and 1 will contract the point cloud while values above 1 will enlarge it. Lastly, the rotation factor, θ, will rotate the point cloud about its center. This factor uses radians, usually from π to -π. If two point clouds are centered at the origin, one can be manipulated by the factors until it most closely 'fits' with stationary cloud.

The process of fitting uses Equation 3.8 to find the manipulated point cloud with the least amount of distance between each point to the stationary cloud. Ideally, the manipulations (with varying, non-overlapping permutations of s and θ) of the first point cloud are saved in memory and each is compared to the second cloud. The respective points in each cloud are subtracted and the 68 differences are summed. The resulting error measurement is also saved in memory and corresponds to the particular s and θ that produced it. When all of the different scalings and rotations have been analyzed, the lowest calculated error will correspond to the point cloud that best ‘fits’ the stationary cloud.

$$E(n) = \sum |(x_2(n) - x_1)| + \sum |(y_2(n) - y_1)|$$

Equation 3.8. Calculating the error between two point clouds (Cootes, Taylor, Cooper, & Graham, 1995).

This algorithm was also used to find the average face in the entire set of point clouds in the Cohn-Kanade database. It was iterated several times to get the best representation. In the first iteration, the first face in the set was used as a stationary cloud and all other facial point sets were fitted to it. After all of the faces were fit, the mean across the entire fitted set of point clouds was computed. This new mean was then used as the stationary set and all of the other faces in the Cohn-Kanade set were fit to it. Again, the mean of these newly fitted point clouds was calculated to get a more accurate representation of their true mean value. The process was repeated 3 more times, until the error between each new mean face was statistically significantly low.

ASM Molder. The purpose of a molder is to bump some points in a cloud into different positions. These different positions are what give shape to different emotions. Once again, the modes of variation attained from a PCA-like process can account for all the differences between shapes of point clouds in the set. This is required because the mean face that the fitter uses is only one expression, and the automatically placed points on a random image will not always be in a neutral configuration.

Prior to implementing the molder, some parameters needed to be optimized. Namely, the number of modes of variation had to be chosen and the degree to which each mode would be acceptably altered. To set these two limits, aspects of the molder needed to be built and tested. Equation 3.9 shows the formula used for obtaining the covariance matrix for the entire set of point clouds.

$$S = \frac{1}{s-1}\sum_{i=1}^{s}\left(data_i - \overline{data}\right)\left(data_i - \overline{data}\right)^T$$

Equation 3.9. Calculating the covariance of the point could data set, where $\overline{data}$ is the mean of that data set. (Cootes, Taylor, Cooper, & Graham, 1995)

Both the eigenvalues and eigenvectors can be computed from the covariance matrix, S. The simplified version of the operation can be seen in Equation 2.4. As previously seen with PCA, the columns of the eigenvectors and eigenvalues are sorted so that the magnitudes of the eigenvalues are in descending order. The first parameter can be calculated at this step: the acceptable number of modes. Since each eigenvalue holds information on how much variance each mode contains, summing the first i eigenvalues and dividing by the sum of all eigenvalues can give a rate of variance for those i eigenvalues. Since almost 60% of the variance can be explained using only the first three eigenvalues, three modes of variation were chosen. As a result, only the first three eigenvectors are used and this new matrix is labeled R.

$$molded \approx \overline{data} + Rb$$

Equation 3.10. Calculation for generating an expressive face using b (the varying eigenvalues) and R (the eigenvectors). (Cootes, Taylor, Cooper, & Graham, 1995).

The final equation for creating the molder is above. It takes into account the mean point cloud, the first three modes of variation in the eigenvectors (which do not change), and a variable vector b. When the elements in b are all 0, then the molded face will be the same as the mean, but as the elements of b start to change, the molded face will start to warp. The last parameter for the automatic point annotator is decided here: the range of values that b should use.

The source paper defines the acceptable ranges for b_i to be $\pm 3\sqrt{\lambda_i}$, where λ_i is the ith eigenvalue. Originally, the eigenvalues describe a certain dimension's variance, so each dimension is given a capped range of values based on the variance. As with the fitter, multiple permutations of b are generated using these constraints. Each permutation was plugged into Equation 3.10 and then their errors in comparison to searched points were calculated using Equation 3.8. The next section will detail what searched points are.

ASM Searcher. The searcher is the last piece of the annotator algorithm. The purpose of the searcher is to push placed points into the correct neighborhood. When the first mean face is placed on an image, it will not be in the correct location and the points will not be positioned optimally. The searcher works by creating a neighborhood for each point and then scanning that neighborhood to see the best location to re-place that point.

Prior to discussing the specifics of the searcher, it is important to understand how the automated facial point annotator works. The following steps are taken:

1. Put a generic mean face point cloud on the image, preferably as close to the actual face as possible.
2. Use the searcher on each point to bump them in the generally correct direction. This generates a new point cloud with slightly more dispersed points.
3. Use fitter tool to fit a new mean face on the newly created dispersed point cloud.
4. Use the molder tool to mold the new mean face to the dispersed point cloud.
5. Use the molded points as the new static point cloud, put a new generic mean face point cloud that will be molded and fit to the new static cloud, and repeat steps 2 through 4 until the annotator converges on a solution.

The process needs a mean face to get started but it is iterative after that. It uses the searcher, fitter, and molder several times and stops when the molded faces do no exhibit significant change from the previous iteration. The searcher generates point clouds that don't always look like faces in the earlier iterations. This is why a new mean face is always inserted and fitted, so the point clouds don't end up diverging into alien-looking faces. The specifics of the searcher will now be discussed.

Rather than using the literature's implementation of a search algorithm, a novel approach was taken. The searcher in this process is based on previously explored technique: DCT/DST patches. The patches were a way to quickly compare two feature sets:

one set that came from the neighborhood of a point and the other set that was representative of the point across all of the dataset's images.

The training set consisted of two elements for each of the 68 points: a mean DCT vector of length 120 and a covariance matrix. To retrieve this information, each point of each face in the Cohn-Kanade database was used as training. Across all faces, the first point was taken, patched, and transformed. The resulting feature vectors for the first point across all faces were then used to compute their mean and to generate a covariance. This process was repeated for the remaining 67 points and resulted in 68 total means and covariance matrices.

The searcher works by taking a neighborhood of pixels around a point and generating a patch for each pixel of that neighborhood, where that pixel is the center of the patch. These patches are also transformed into DCT vectors of length 120. As a result, each point generates *N* feature vectors, where *N* is the area of the neighborhood (the total number of pixels in that neighborhood that generate the patches). Since each point also had a mean, $\bar{g}$, and covariance, Sg, that was previously calculated in the training set, each feature vector in the neighborhood can be compared to those values to see how closely they resemble the true location for where the point should be placed. The patch that generates the lowest error in the neighborhood signifies that the point should be moved to that specific patch's center since it most closely resembles patches from the training set. Equation 3.11 shows the error calculation.

$$E(N) = (\bar{g} - g(N))Sg(\bar{g} - g(N))'$$

Equation 3.11. Calculation for finding the error between a patch, g(N), and the training set.

Classification

The goal of classification is to label unknown incoming inputs based on previously observed inputs. This requires the data to be split into a set that trains the classifier and a set that tests the classifier. A classifier will only label the input as a class that it was trained on, so the expected output for this research was one of the seven emotions present in the Extended Cohn-Kanade set.

Linear Discriminant Analysis. Linear Discriminant Analysis was one method used in this research to perform emotion classification on facial images. There was a MATLAB function that allowed for minimal coding of the classification algorithm, but the philosophy behind it will be discussed now. To perform linear discriminant analysis there are several concepts that need to be understood: posterior probability, multivariate normal density, and cost.

The only required inputs for the ClassificationDiscriminant.fit() function in MATLAB are the training samples (arranged in a matrix where each row is a separate sample) and their respective labels (where each row is the label for the corresponding sample). In simpler terms, LDA functions by drawing lines between the classes in an attempt to separate them. It later uses these barriers to best predict labels for incoming test points in the predict() method.

The predict() method produces a label for a test point using the model generated by ClassificationDiscriminant.fit(). Specifically, the equation that predict() implements to arrive at the concluding label is seen in Equation 3.12. In that equation, $\hat{P}(k|x)$ refers to the posterior probability of class k for observation x, and $C(y|k)$ is the cost of classifying an observation as y when its true class is k. While the former will be discussed soon, the latter is a simple condition: the cost is 0 if the class of y is the same as class k and it is 1 when they are different. The predicted label will be the one that has the lowest classification cost.

$$Label = \arg \min_{y=1,\ldots,K} \sum_{k=1}^{K} \hat{P}(k|x)C(y|k)$$

Equation 3.12. Formula used to compute the label for a test point using the predict() method in MATLAB for discriminant analysis. (Discriminant Analysis, 2015)

The equation is run K times (where K is the number of classes) since the minimum of those runs will be the resulting label. At each run, there are K-1 posterior probabilities that are summed up. Although the summation sign suggests K posterior probabilities to be summed up, at one point, the y value and the k value will be the same causing $C(y|k)$ to equal 0. In all other cases, $C(y|k)$ is 1, so essentially only the posterior probabilities are summed up. Each run considers a class for the unknown test point and then checks how far away it is from the other classes. The lower the value, the farther away it is from all other classes.

The next component to discuss is the posterior probability. Equation 3.13 displays the computation of posterior probability that MATLAB uses. It is the combination of the multivariate normal density $P(x|k)$, the prior probability $P(k)$, and the normalization constant $P(x)$. The prior probability, much like the cost, is a simple computation. In the case of a uniform prior probability, it is a value of 1 over the total number of classes. If the empirical prior probability was used, it would be the number of samples for class k divided by the total number of samples used during training, although the uniform calculation is used by default. The normalization constant is similarly easy to compute: it is the sum over k of $P(x|k)P(k)$.

$$\hat{P}(k|x) = \frac{P(x|k)P(k)}{P(x)}$$

Equation 3.13. Formula used to compute posterior probability (Discriminant Analysis, 2015).

The multivariate normal density is perhaps the most complicated computation made during discriminant analysis prediction. It is detailed in Equation 3.14 and is responsible for calculating a test point's relation to the class distributions. The equation is used for a point with respect to each class, and when the result is higher, there is a higher probability that the point belongs to the class it is being computed for. If the result is lower, then the point is not as likely to be part of that class's distribution since it is technically farther away. In the equation, x denotes the test sample feature vector for which the label is being predicted. It is subtracted by the mean of all training samples for a class k (denoted as μ_k). Σ_k also appears in several areas of the equation and it represents the covariance matrix of the training points of class k. However, $|\Sigma_k|$ is the determinant of the covariance matrix while Σ_k^{-1} is the inverse of the covariance matrix.

$$P(x|k) = \frac{1}{(2\pi|\Sigma_k|)^{\frac{1}{2}}} e^{(-\frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k))}$$

Equation 3.14. Formula used to compute multivariate normal density (Discriminant Analysis, 2015).

MATLAB offers some flexibility when it comes to Discriminant Analysis, presenting options for both linear and quadratic discriminant analysis. Conceptually, the key difference between the two approaches is how the classes are divided. Straight lines segregate the classes in LDA while conic sections like parabolas and ellipses separate the classes in QDA (quadratic discriminant analysis). Mathematically, QDA uses a slightly different formula than LDA for the posterior probability.

Aside from the predicted label of the input test sample, predict() can also output the certainty score for each of the classes. These scores were normalized from 0 to 1 and whichever was the highest was used as the label for the sample. Although it is not necessary to further investigate these scores after the classification has been made for a simple LDA system, they can often hold some clues on how to push the system accuracy even further. This approach is discussed in the future section, Section 3.9.3.

The scores were also used when patches and full faces were mixed together for training and testing. In the training phase, the full faces and patches were sent en masse to generate the math models for the classifier; only the labels mattered here. However, the testing phase was conducted with more attention. As each test sample entered the predict() method, it essentially had a 2D matrix of features, where the first row was a feature vector for the full facial image and the preceding rows were patch feature vectors. The result from predict() produced a 2D matrix as well, where each row contained the class scores for the respective feature vectors (full face and patches). To get the final label of the input test sample, each column (one for each emotion) was averaged. This operation produced a single row of scores in the end, the high of which was used as the conclusive score for the test sample.

Kth Nearest Neighbor. The kth Nearest Neighbor algorithm is perhaps one of the simplest classification algorithms available. The ease of understanding and implementing it were factors for choosing it for this research. This classification algorithm operates by essentially treating each feature vector (all training samples) as a point in N-space, where N is the length of the feature vectors. When a new feature vector (a test sample) is added, the distances between that sample and all other points are computed. There are several different distance calculations available including: Euclidean, Mahalanobis, and Chebychev. These distances are ordered from smallest to largest, and the top k (a user defined variable) are picked. The assumption at this step is the new feature vector will most likely have a label that is the same as the points in its proximity. Therefore, the labels of the top k closest training samples are collected and the most common (the mathematical mode) label is used as label for the test sample. An odd value for k is used to eliminate tie events.

There are two variable aspects to kNN: the value of k and the distance metric. Experiments were run with the k value changing each time. Since odd values of k are used, the tested values were 1, 3, 5, and 7. In terms of distance measures, Euclidean (which is expressed in Equation 3.15) was chosen. Since performing extra experiments to deduce which distance measure was most optimal would have used up more time, Euclidean was chosen for its ubiquity and to save that time.

$$d(X_{tr}, X_{te}) = \sqrt{(X_{tr1} - X_{te1})^2 + (X_{tr2} - X_{te2})^2 + \cdots + (X_{trN} - X_{teN})^2}$$

Equation 3.15. Formula to compute Euclidean distance between to vectors (X_{tr} being a training vector and X_{te} being a test vector) that are N dimensions long.

Similar to the LDA output, a test sample has a confidence score associated with each class that kNN was classifying for. For each class, the score is equal to the amount of neighbors of that class divided by the total number of neighbors being evaluated. If three out of the five nearest neighbors are labeled with neutral, then the neutral score for the test sample would be 0.6. The class with the highest score is used as the final classification for the test sample.

Experimental Procedure

Due to the high number of experiments that needed to be performed, it was crucial to have standardized methods of execution and evaluation. This section will discuss the way each experiment was performed and how the accuracy was derived and compared to previous runs. The methodology listed here allowed system tests to be highly repeatable and offer valuable information on how components in the system were behaving. The system's accuracy was an aggregation of several different values that originated from how the data and results were organized.

Data Partitioning. The purpose of Data Partitioning was to equally divide the data into blocks of testing and training sets. The input to this stage required that all of the features had been extracted and added to the IDM along with their labels. After this was created, the next step was to pick a value for N, the number of blocks that the data should be divided into. The blocks needed to be roughly the same size along with the same number of samples from each class. With this setup, the first block was used as the testing set while the remaining N-1 were the training set. A value of 7 was used for N to create approximately a 20/80 split in the data, where 20% of all of the data was for testing and 80% was for training.

The experiment did not end with the classification of the samples in the first block. The next block was used as the test block while the remaining N-1 blocks were training. This procedure was repeated N times, where each block was eventually used as a test set once while the remaining blocks were used as training. Data partitioning served to cross-validate the data and get a comprehensive sense of the system accuracy. It is also important to note that each time a new instance of the system ran, the IDM was randomly permutated resulting in different blocks between runs. The blocks still had consistent sample sizes of each class contained within, but the samples themselves were randomly chosen each time. The system therefore did not produce the same results each time and could be run multiple times to get more aggregate accuracies.

Top-N Ranking. The confidence information from classification algorithms like kNN and LDA contain valuable information that can be exploited to further increase the accuracy of an emotion recognition system. Each classification has some confidence associated with it – a value between 0 and 1 where higher values represent higher confidence. Since the algorithms model multiple classes at once (rather than doing a one vs. all approach) then each class has a confidence score associated with the test sample in question. The class with the highest score is the one that is used as the label for the sample. However, some experiments were run to see how often the correct label was in the top N results of the scores.

The key alteration necessary to perform the Top N experiments was to register classifications as correct if the right label was in the Top N (a user-defined variable) confidence scores for a test sample. The scores were ordered from lowest to highest and the top N were chosen. At first, N was equal to 1, so only the highest score was picked (thus no change to the recognition system) but the value of N was increased by 1 in each run. When N was two, the two highest scores were examined, and if the correct answer was between them, the classification was counted as a success. At the end of each run, the overall system accuracy was calculated and stored in a table.

The motivation behind performing the Top N experiments is to understand how 'close' to the real answer the classification systems were. In many cases, the disparity between the top two scores was very low, and could have arrived at the correct answer given some extra information. It was posited that if the correct answer was amongst the top 3 highest confidences at least 85% of the time, then it was worthwhile to start using patches to further bump the scores into producing the correct classification. The Top N experiments were run on a recognition system that only used full face images. The success of the experiments (detailed in Section 4.2.6) was what actually led to the investigation of facial patches to supplement full faces for emotion classification.

Feature Fusion. One of the stated goals for this research was to explore the effects of fusing multiple feature sets into one system that classified emotions. There were many ways that this problem could have been met, but ultimately, only one was implemented. This decision came from the conclusion drawn from previous work. Out of the three decision-level feature fusion methods (borda count, decision voting, and score averaging) the latter was chosen. Score averaging generated the highest results when it was tested in a previous work (Smirnov, Muraleedharan, & Ramachandran, 2015).

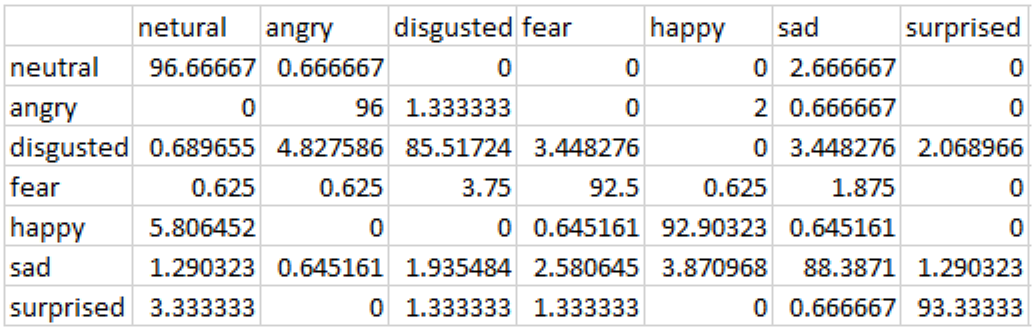
The implementation of score averaging feature fusion was very similar to how local features (frames) were added to global features (face only). The basic premise involved the aggregation of scores from separate classifiers. It was expected that each classifier produced a standardized output: a row vector of length 7 where each value was the likelihood that a test sample belonged to that particular class (7 dimensions for 7 classified emotions). More importantly, the values in the row had to add up to a value of 1 as a normalization measure. Since the classifiers (both LDA and kNN) met these criteria, their output scores were fused by concatenating their output class scores vertically and averaging along the columns. The resulting averaged row vector was also of length seven, and the element with the highest value was used as the final classified emotion.

In a system that used feature fusion, each classifier saw its own training data but the testing samples were synchronized. The same input image was fed into all of the classifiers to produce the scores to be fused.

Calculating System Accuracy. The labels of all the samples were known for the duration of the research. Even though the goal of classification was to classify input data, the answer was already known since all samples in the Extended Cohn-Kanade and JAFFE database were labeled. The results from the classification algorithms could then be compared to the actual labels from the databases to see if they were classified correctly. As a result, the accuracy of the system could be assessed with these two pieces of data.

However, there are several steps required before calculating the overall system accuracy. Along the way, different kinds of accuracies are averaged together to obtain the overall accuracy. The first of such accuracies is known as the 'fold accuracy.' As previously stated, the data is partitioned into N blocks and run N times. Each run is a 'fold' because the method of data partitioning is formally known as 'k-fold cross-validation.' To better understand how accuracy is tabulated and assessed, it is first necessary to understand confusion matrices.

A confusion matrix is a helpful tool that stores classification data. It is an N by N grid, where N is the total number of classes. Each row and each column represents one class, where the rows are actual classifications and the columns are predicted classifications. Refer to Figure 3.8 for a visual representation of this. After each input is labeled by the classifier, a value is added to the confusion matrix. First, the row that corresponds to the actual label value of that input is used and the classifiers predicted label is used as the column value. At the intersecting cell, a value of one is added to whatever the value was previously.



	netural	angry	disgusted	fear	happy	sad	surprised
neutral	96.66667	0.666667	0	0	0	2.666667	0
angry	0	96	1.333333	0	2	0.666667	0
disgusted	0.689655	4.827586	85.51724	3.448276	0	3.448276	2.068966
fear	0.625	0.625	3.75	92.5	0.625	1.875	0
happy	5.806452	0	0	0.645161	92.90323	0.645161	0
sad	1.290323	0.645161	1.935484	2.580645	3.870968	88.3871	1.290323
surprised	3.333333	0	1.333333	1.333333	0	0.666667	93.33333

Figure 3.8. Typical confusion matrix.

Visually inspecting the above figure, it can be deduced that the diagonal values represent correct classifications while all other cells are misclassifications. Once again, this is because the diagonal cells represent inputs that were classified with the same label as their true label. The confusion matrix provides information on both the true positive and false positive detection rates. Dividing all cells in a row by the total number of samples of the given class generates percentages, which are the basis for accuracy values in this system. The overall fold accuracy can be calculated by averaging the class accuracies (the diagonal values).

After the fold accuracy comes the trial accuracy. One trial is equivalent to N folds, and the trial accuracy averages all of the folds together. To further verify the accuracy of the system, trials were run five times, resulting in a single 'system accuracy.' The system accuracy contains the averaged accuracies from the five trials. It can either be a single value known as the Overall Detection Rate or as N values (where N is the number of classified emotions), which is the System Class Detection Rate. When accuracies were stated broadly in this thesis, they were usually refereeing to the Overall Detection Rate.

Chapter 4

Results and Discussion

Basic Approach to Results and Discussion

Much of the progression of this research was based on a central idea: experiments that yielded higher accuracies were explored further and added to the overall system. The result was a system that combined many finely tuned and successful elements. This chapter explores which decisions were made and, more importantly, why they were made.

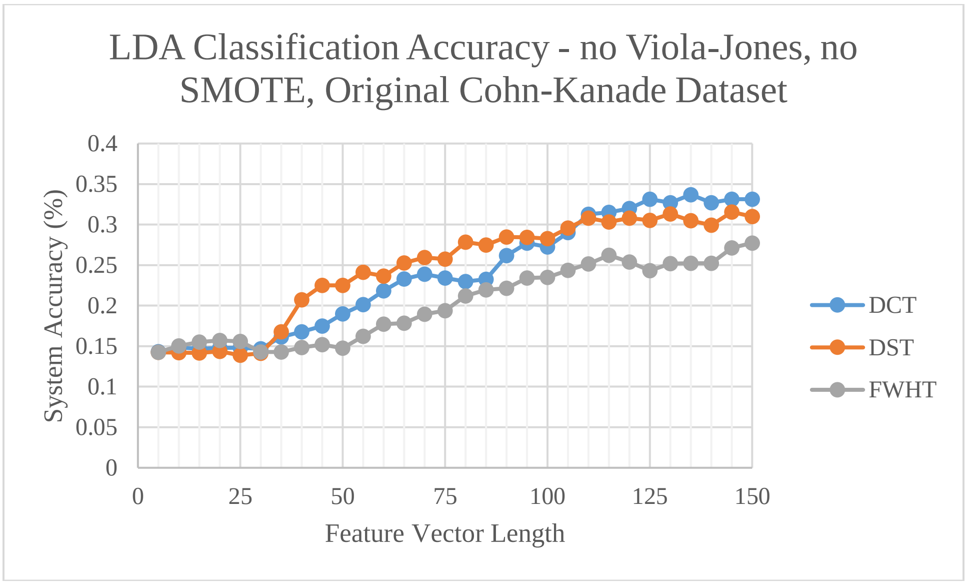


Figure 4.1. Initial set of experiments run on the original Cohn-Kanade Database using an LDA Classifier.

The preliminary results, see in Figure 4.1, did not look very promising. The basic layout for experiments can be seen here and will now be discussed before proceeding. A single experiment consisted several previously discussed steps. First, different pre-processing measures used on the input images to prepare them for feature extraction. Next, the feature extraction algorithm of choice was applied and retrieved a feature vector of some specific length to be used in a classifier. The classifier of choice ran the partitioned data through and kept track of average accuracies. After the 7 folds and 5 trials were complete, a single system accuracy could be determined for that particular experiment. Figure 4.1 displays a set of experiments, where different feature extraction algorithms (DCT, DST, and FWHT) and feature vector lengths (increments of 5 from 5 to 150) were attempted. This set of experiments will be the most common amongst most of the results, only differing by factors such as: pre-processing steps, SMOTE ratios, dataset considerations, and classifier.

Cohn-Kanade Dataset Results

Most of the experiments were performed on the Cohn-Kanade database first. The different pre-processing, feature extraction, and classification choices were tested on this database and it was assumed many of the same results would be mirrored in experiments performed in the next database: JAFFE. The following sections will detail each decision that was made and how it impacted the emotion recognition system.

Case for Viola-Jones. In this initial set of experiments, the raw Cohn-Kanade images were used as input, there was no pre-processing step, no Viola-Jones facial detection separated the face from the background, and the LDA classifier was used. SMOTE was also not used to even out the disparate sample sizes amongst the emotions. As a result, even the best performing experiment (DCT feature vector of length 135) did not get the answer right even 35% of the time.

The same set of experiment was carried out again, with one difference: Viola-Jones was applied to each image to isolate the face for feature extraction. The results can be seen in Figure 4.2 below.

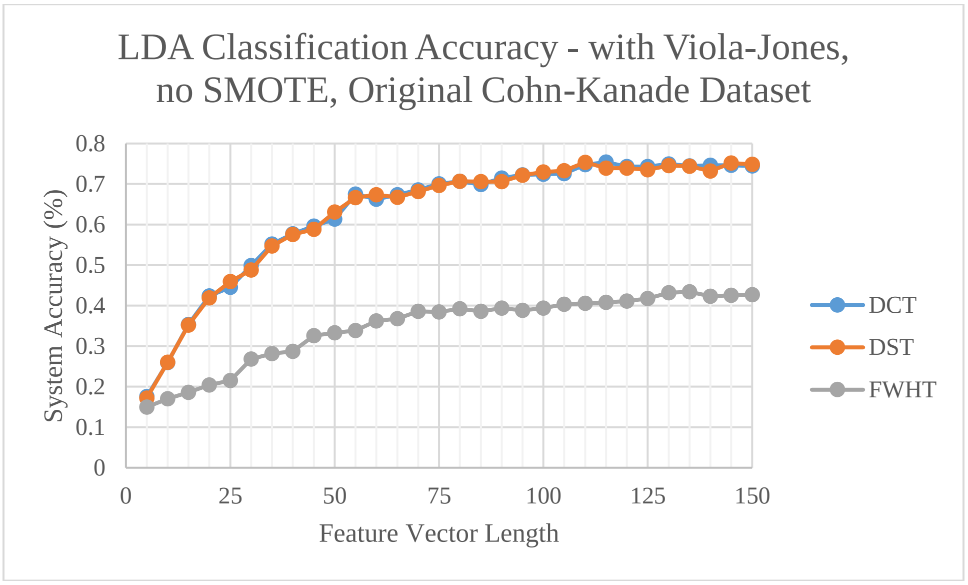


Figure 4.2. Set of experiments detailing results with Viola-Jones applied to each input image.

The system accuracy for both DCT and DST more than doubled for all of the feature vector lengths. The best performing features were around 105-110, at a system accuracy of 75%. FWHT features, on the other hand, did not see such a drastic boost in accuracy with the implementation of Viola-Jones. Regardless, this set of experiments demonstrated the importance of isolating the face from the rest of the picture. Otherwise, elements in the background will contribute noise to feature vectors and lead to a greater number of misclassifications. All experiments carried out after this one used Viola-Jones.

It is also useful to point out that using feature vector lengths about 150 was not necessary. This is evident by the plateau of accuracies after a size of roughly 100 dimensions. This plateau will be present in most of the preceding Figures, showing that virtually any dimension size past 100 will produce the same accuracy. Due to this, 120 is a commonly used vector length for single experiments that require a lot of time to perform (rather than performing the entire set of 30, starting from 5 and going to 150 in increments of 5).

Table 4.1

Breakdown in sample size after Cohn-Kanade Database was processed with Viola-Jones.

index	emotion	Original sample size	Viola-Jones sample size
0	neutral	526	463
1	anger	41	39
2	contempt	0	0
3	disgust	41	34
4	fear	19	18
5	happiness	61	52
6	sadness	22	20
7	surprise	75	73
8	unclassified	267	241

During the process of isolating faces from the facial images in the Cohn-Kanade set, a small discrepancy occurred. Not all of the faces were captured using Viola-Jones, and as a result, the original sample sizes for the emotions was no longer the case. Each emotion was negatively affected by losing a few samples. Rather than having 1052 total samples, only 940 remained. This means that wherever Viola-Jones was applied to the

Cohn-Kanade database, the new sample size breakdown seen in Table 4.1 was the actual breakdown.

Case for Shrunken Images. The isolated faces from the Viola-Jones algorithm were not uniform in size. Depending on how far away the subject was from the camera, and their physical face size, the isolated results came in all different 2D matrix dimensions. A pre-processing step of scaling was added to normalize the height and width of all the cropped faces to 128 by 128 pixels. This was an arbitrary value, but to see the effects, other values were also tried: 64 by 64 and 32 by 32 pixels.

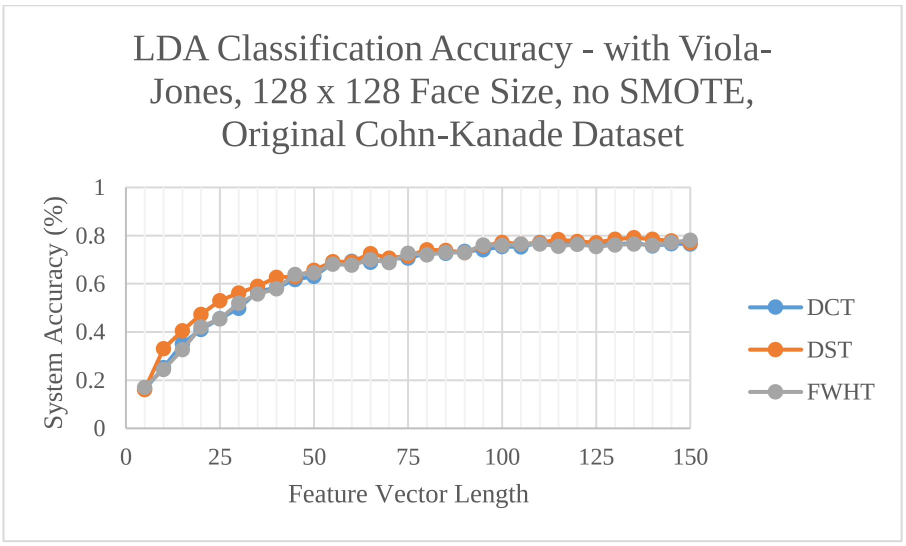


Figure 4.3. Set of experiments detailing results with facial images scaled to 128 x 128 pixels.

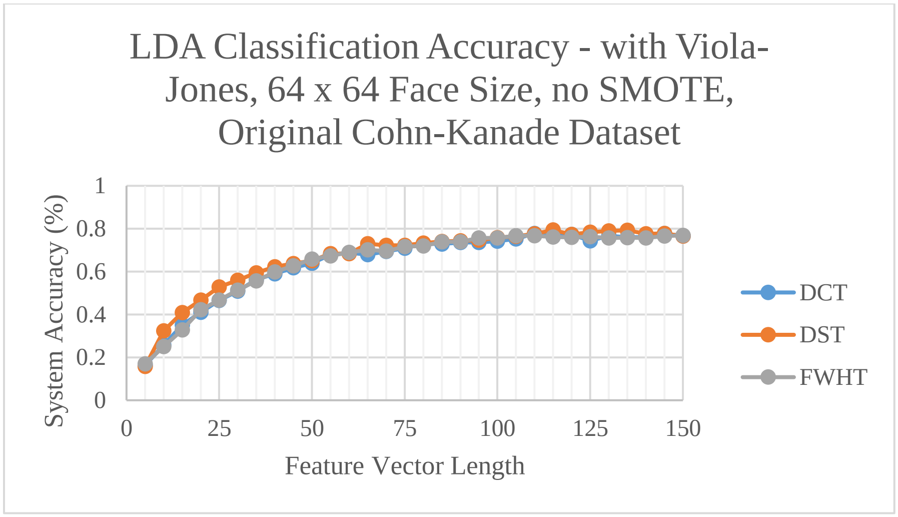


Figure 4.4. Set of experiments detailing results with facial images scaled to 64 x 64 pixels.

A curious effect of scaling the facial images down to 128 by 128 and below was seen. Previously, FWHT feature vectors did not break 45% accuracy, but the scaling suddenly made the results comparable to both the DCT and DST. The accuracy was boosted for all three cases compared to not scaling, where accuracies floated around 76%-79% for well performing vector lengths. This was most likely caused by the FWHT preference for square input matrices of size 2^n by 2^n. With that size requirement fulfilled, no zero-padding was necessary for the faces and the faces retained their original information.

32 by 32 results were more of the same. The results did not get any better or any worse compared to the other two scaling. A visual inspection of the 32 by 32 facial images seemed to show a significant loss of quality, but the frequency-based feature vectors were able to maintain their accuracy at such pixelated levels. Future experiments were carried out with 64 by 64 scaling.

Case for SMOTE. It is important to recall Table 3.1 to get a reminder of how unbalanced the original dataset was in terms sample sizes for each emotion. Since each emotion was a set of still images from a video of a subject going from a neutral pose to a posed expression, the number of neutral samples was equal to the sum of the other emotions. SMOTE was implemented to synthesize extra samples for all other emotions to bump their total available samples to match neutral's. The assumption was that a more balanced dataset would produce higher accuracies because classifiers could separate the classes more accurately.

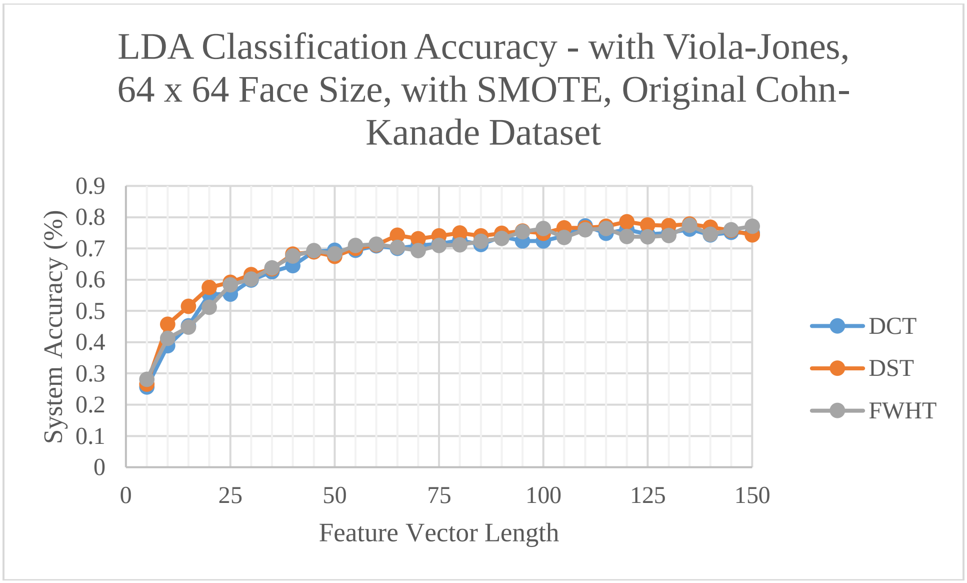


Figure 4.5. Set of experiments detailing results with full SMOTE.

Given Figure 4.5, it seems as if there was no appreciable change compared to not using SMOTE. The results still do not break past 80% system accuracy. However, the class average accuracies were also examined. The system accuracy is an average of all the class averages and doesn't reveal any information on which emotion is classified better or worse. The class averages for this set of experiments will be detailed in Section 4.2.4 and in Figure 4.8. Before exploring those, some more sets of experiments will be discussed.

Table 4.2

Breakdown in sample size for each emotion after full SMOTE is applied.

index	emotion	Viola-Jones sample size	SMOTE sample size
0	neutral	463	463
1	anger	39	390
2	contempt	0	0
3	disgust	34	476
4	fear	18	396
5	happiness	52	520
6	sadness	20	400
7	surprise	73	438
8	unclassified	241	241

The new breakdown of sample sizes for each emotion can be seen in Table 4.2. In the worst case, the 18 samples of fear had to be inflated by 22 times its original size. Since so much synthetic data was being generated during SMOTE, a slightly different sample distribution was tried that cut down on interpolated data. Table 4.3 details a distribution that cuts the neutral sample size by half and bumps the rest of the samples to that new maximum. Figure 4.6 shows the results of running a set of experiments with the 'half SMOTE' sample size regimen.

Table 4.3

Breakdown in sample size for each emotion after half SMOTE is applied.

index	emotion	Viola-Jones sample size	SMOTE sample size
0	neutral	463	232
1	anger	39	195
2	contempt	0	0
3	disgust	34	238
4	fear	18	198
5	happiness	52	260
6	sadness	20	200
7	surprise	73	219
8	unclassified	241	241

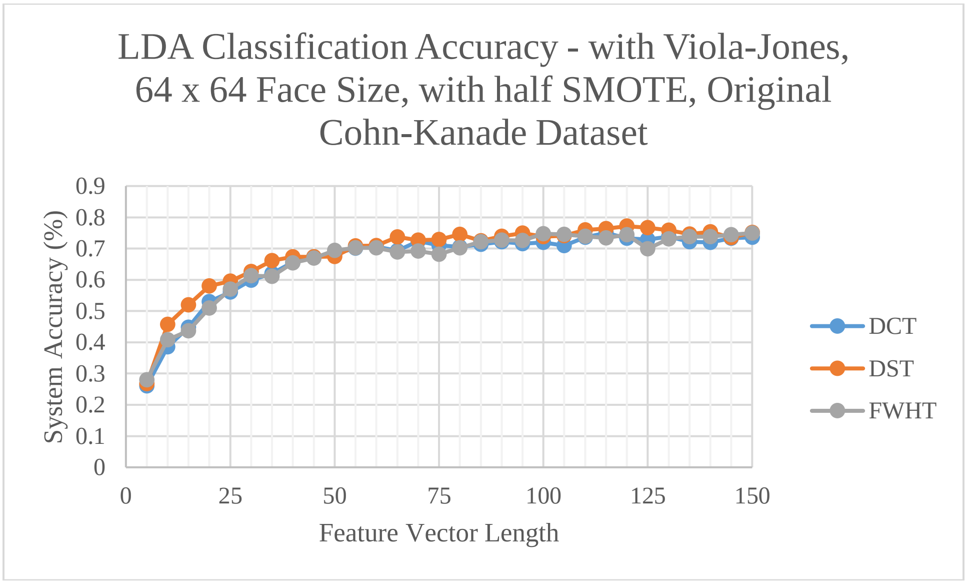


Figure 4.6. Set of experiments detailing results with half SMOTE.

The half SMOTE data regimen did not produce any higher results. In fact, accuracy amongst all three feature sets, across all feature lengths, was a few percentage points lower. The takeaway finding from this set of experiments was that it is better to have more data for training and testing than less. Even if the dataset features high amounts of synthetic data, the discriminatory functions in LDA perform better with larger quantities of information.

Further Increasing the Natural Sample Sizes for Each Emotion. Two factors contributed to the decision of adding extra samples to the Cohn-Kanade dataset that was used for experiments. First, it was noted that some of the labeled emotions did not appear to be correct during visual analysis of training samples entering the classifier. Secondly, there was a large amount of unused data in the dataset. This data had a classification label of 8, and was always discarded during the experiments. In this set of experiments, the already-existing samples were double-checked and corrected if they were misclassified. The unused samples were also visually scanned and given classification labels if they met FACS criteria for a basic emotion. The new breakdown for sample sizes after this process are shown on the next page.

Table 4.4

Breakdown in sample size for each emotion after classification revision and extra samples were added.

index	emotion	Viola-Jones sample size	Extra sample size	SMOTE sample size
0	neutral	463	463	463
1	anger	39	40	440
2	contempt	0	0	0
3	disgust	34	34	476
4	fear	18	31	465
5	happiness	52	63	504
6	sadness	20	40	480
7	surprise	73	75	450
8	unclassified	241	194	194

The two emotions that previously had the lowest amount of samples benefited the most from this dataset revision. Fear and sadness were significantly under-represented and SMOTE had to generate much more synthetic data for them as a result. The new samples allowed for lower SMOTE multipliers and more representative synthetic samples. Anger and disgust did not see too many new samples, however. It was hard to distinguish between the two emotions using a manual visual analysis of unused samples. Moreover, many of the unused samples were blends of emotions, so their label could not be set for any one particular classification. In the end, the sample overhaul produced a net benefit to the emotional classification system, as Figure 4.7 shows below.

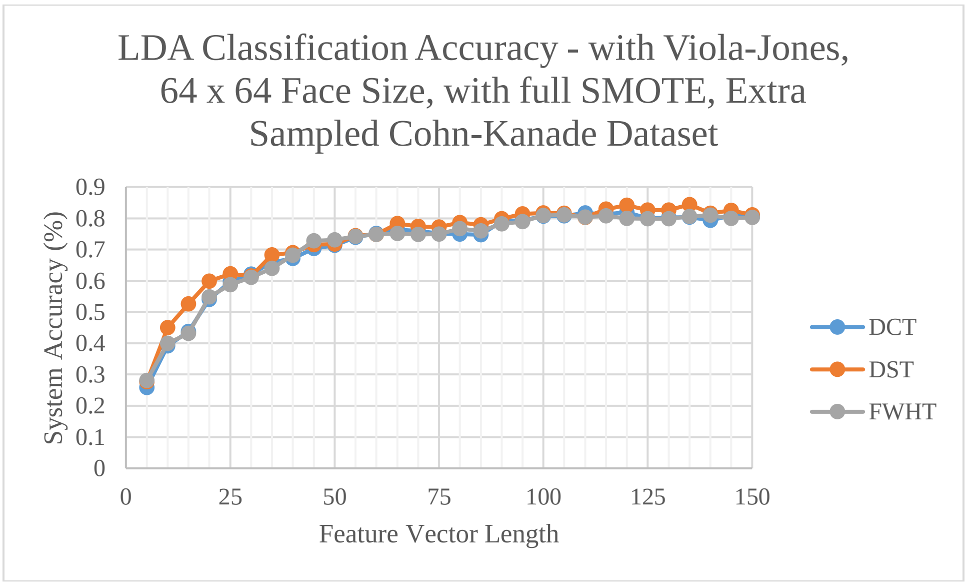


Figure 4.7. Set of experiments detailing results with extra samples.

With this set of experiments, the system accuracy was finally above 80%, and topped out at 84.3% with a DST feature vector of length 135. As the SMOTE tests previously showed, more samples mean better class segregation. This new set of experiments further pushed that point by suggesting more real-world samples are vital to create higher system accuracies.

This was the last set of experiments that dealt with global features using frequency-related transforms. What needs to be examined at this time are the class averages for some of the previous experiments. While SMOTE did not make noticeable difference to the system accuracy, the individual accuracies for each emotion class may have been altered. Figure 4.8 was generated by using DST feature vectors of length 120 and the respective parameters detailed in the legend.

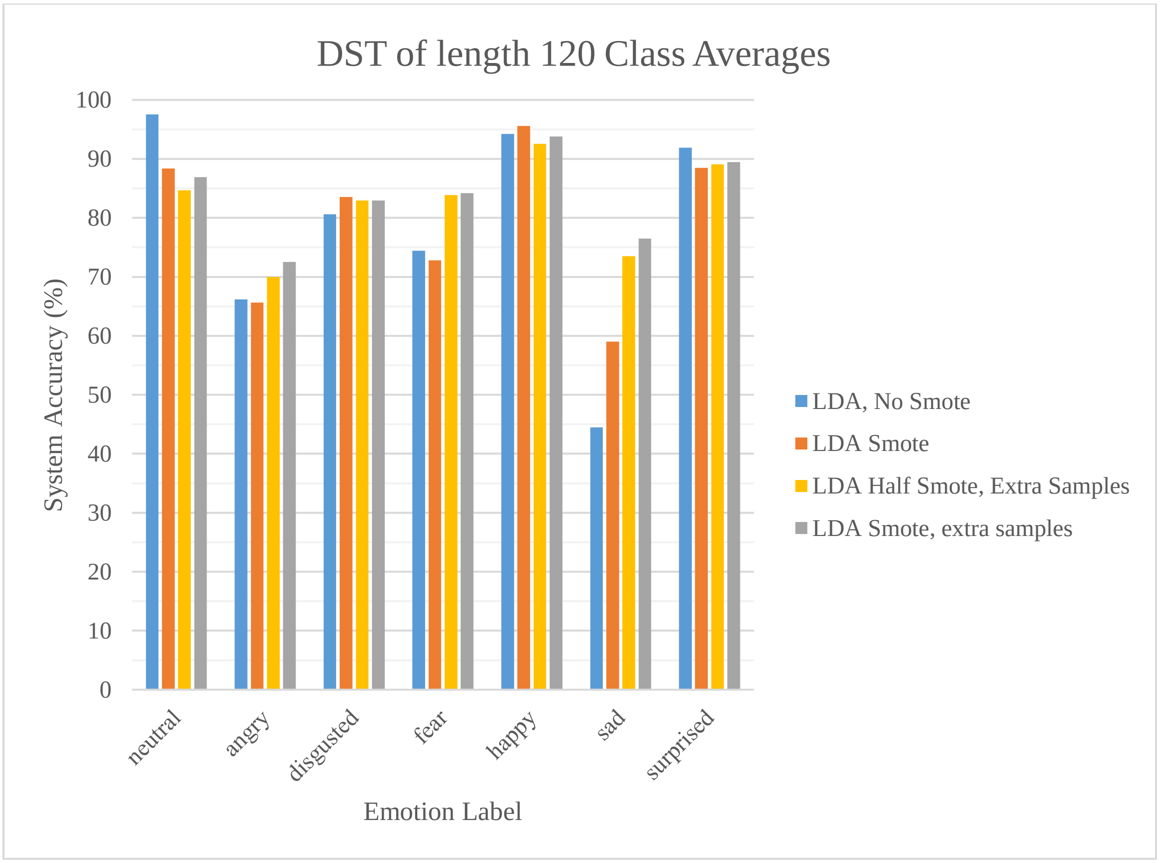


Figure 4.8. Set of experiments detailing results with extra samples.

In the no SMOTE regimen, the class average accuracies were representative of the sample sizes. Neutral had the highest accuracy because the system was trained heavily to recognize neutral compared to the other emotions. Emotions like anger and sadness were harder to distinguish from neutral expressions and they were often misclassified as neutral as well. Emotions like happiness and surprise also showed higher accuracies because of their higher distinction from a neutral expression. Unlike any other emotion, teeth were most often on display in happy samples, and wide open eyes/mouth were most often on display in surprised samples.

The addition of SMOTE mainly increased the accuracy for sadness at the expense of a few percentage points from other emotions. This can be expected since there were technically more points for the classifier to use for class segregation. Even though some of the emotions were negatively impacted in terms of accuracy, it is more important for the classes to have an even distribution of accuracies rather than having a few with high accuracies while the other were significantly lower.

This equity of class averages was achieved even further when extra natural samples were added to the dataset. The additional information went a long way in creating better class boundaries in the LDA classifier. Anger, fear, and sadness saw a significant increase in accuracy despite still being under-represented. With this particular regimen, the accuracies for each emotion were above 70%; the best performing set of accuracies from tests so far discussed.

Lastly, the half SMOTE setup underperformed with all classes. In the case of the figure, the half SMOTE was applied to the extra-sampled data and was compared to the full SMOTE version of that data. It did not boost performance for any class, and showed a small negative percentage difference in all cases.

Frequency Features with KNN Classifier. Similar experiments were also performed using kNN classifiers with k values of 1, 3, 5, and 7. Since previous cases had already laid it out, these experiments included Viola-Jones isolated faces that were cropped to 64 by 64 pixels, full SMOTE, and extra samples. The DST features were the best performing in previous experiments, so only they were evaluated in this new set of experiments to save on time.

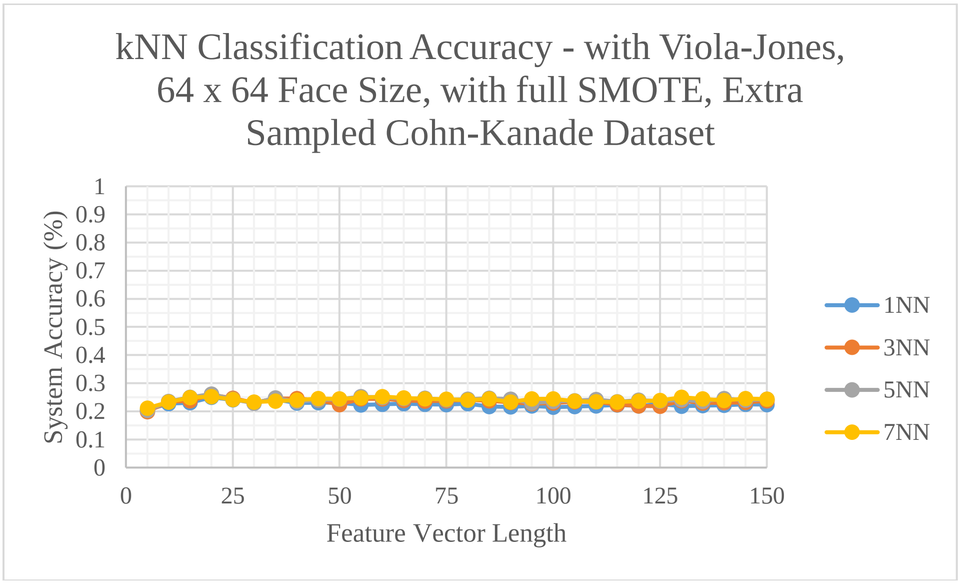


Figure 4.9. Set of experiments detailing results with DST features of length 120.

The figure above shows that kNN did not perform well with DST features. This was the case for all of the frequency-based features (DCT and FWHT). Oddly, there is a peak accuracy at feature vectors with length 20, and then a plateau for vectors of longer length. The neighborhood size did not make a difference, as all permutations performed the same.

Further Understanding of Accuracy. The results shown so far have illustrated only the performance of the system for the most probable answer. The closer that an LDA score for a class approaches a value of 1, the more likely the test sample belongs to that class. Systems up to this point only used the highest scoring class to make an assessment, but a set of experiments were also carried out to consider how ‘close’ the system was to the real answer. If the top 2 highest scoring classes were considered instead of just the single highest, would the correct label show up between the two? How often would it show up in the top 3? If the classifiers were able to compute the correct label in these cases, it would provide a good argument for continuing to explore feature extraction from the face (most notably from local facial features).

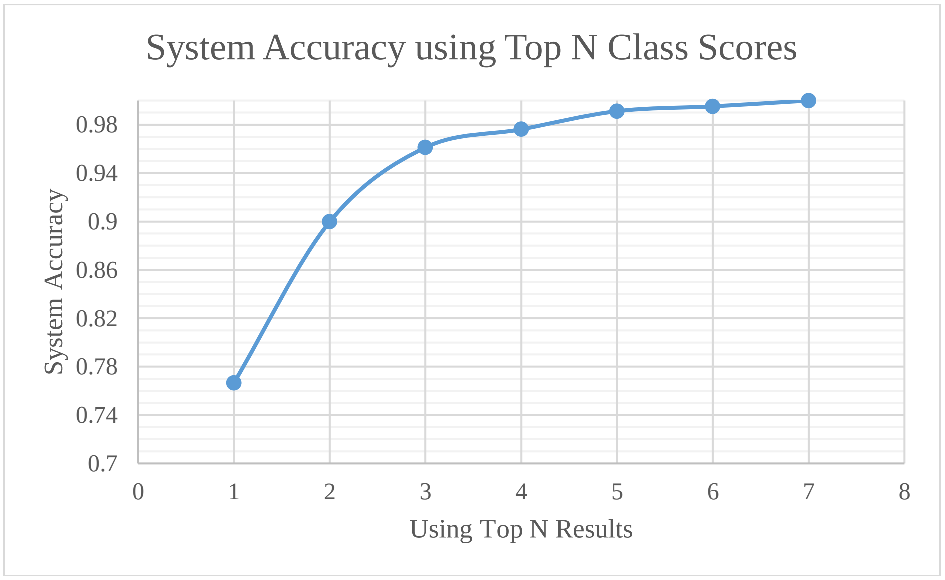


Figure 4.10. Experiments detailing the results of using the Top N class scores for classification rather than Top 1st.

The Top N experiments were run on an LDA classifier using DST vectors of length 120. Since this experiment was performed earlier in this research, the faces were only shrunken to 128 by128 pixels and there were no extra samples present in the Cohn-Kanade Dataset at that time. SMOTE was still applied, but in the older proportions. This explains the noticeable 10% difference in accuracy for the Top 1 test in Figure 4.10 when compared to previous high-performing experiments.

However, the general trend in increasing accuracy is still relevant. When taking only the top 2 results into account, the correct label was present 90% of the time. This is compared to only 76.6% when the top 1st class is examined. In the top 3 results, the correct answer was present 96.1% of the time. As more classes were added to the Top N ranking, the results eventually tapered off to 100% since the correct answer would always be one of the 7 classes.

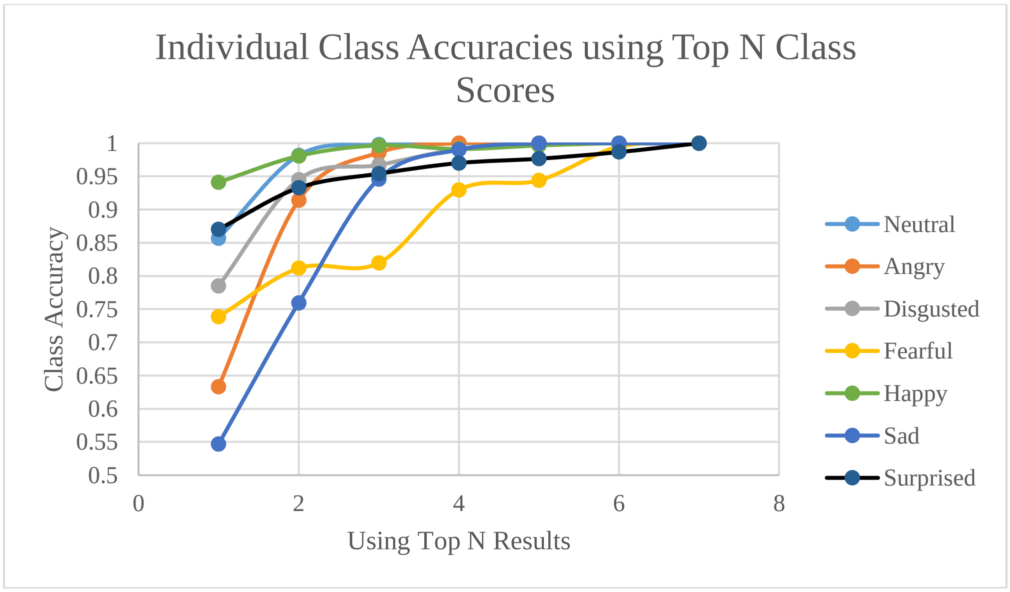


Figure 4.11. Set of experiments detailing the results of using the Top N class scores for classification rather than Top 1st.

The same trend in Figure 4.10 is reflected in Figure 4.11. In the latter, individual class accuracies were examined as the value of N in the Top N was increased. If the actual correct label was a neutral classification, and neutral was amongst the Top N results, it was counted as a true positive result. The class results for Top 1st mirror those of previous experiments. An appreciable rise in class accuracies occurred across all emotions going from the Top 1st to the Top 2 highest scoring classes. The same was true from moving from the Top 2 to the Top 3 with the exception of fear. This suggested that fear potentially would still struggle to be classified if extra information was supplied to the classifier. However, after the sample sizes for each emotion were tweaked, that discrepancy was diminished. Figure 4.12 details the updated results which feature 64 by 64 pixel sized shrunken facial images and extra samples.

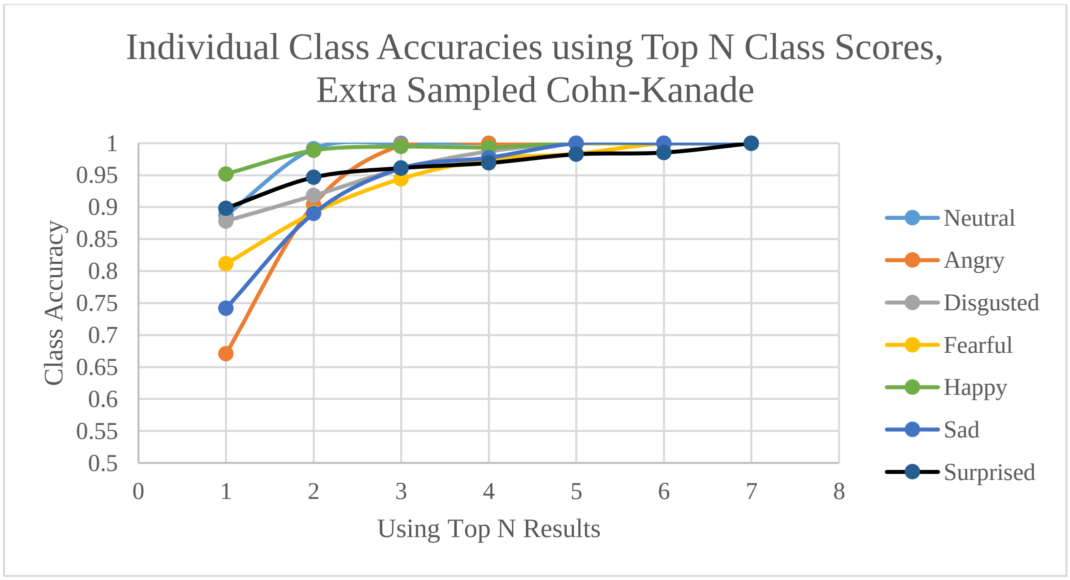


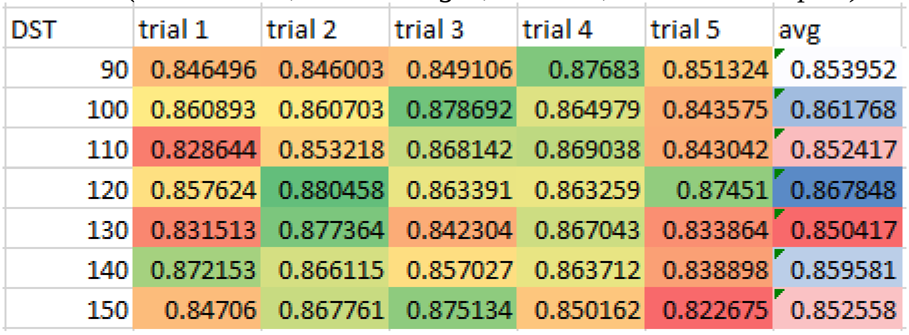
Figure 4.12. Set of experiments detailing the results of using the Top N class scores for classification rather than Top 1st with extra samples.

Such results indicated that the classification system had the potential to do better if some additional information was available to it. The full faces contained most of the discriminatory information needed for LDA to correctly classify an emotion. The logic was that perhaps local features from the same face (like lip corners displaying neutral or exaggerated creases from happiness or inner eyebrows coming together for an angered expression) could add enough unique feature information to push the scores in the correct direction. The next section discusses the results after global and local features were combined to produce a hybrid emotion classification system.

Case for Local Features. In total, 12 local features were taken from each face. They came in the form of patches of pixels and those patches were processed with frequency transforms to create feature vectors. To lower the impact of wildly different feature vectors between global and local features, each respective group was passed through their own LDA classifier and their scores were combined at the output. The aggregate of the scores across each class were averaged to get a final score of those classes per test sample (a test sample being it's global and local features). The highest score from these averages was chosen as the label for that sample.

Table 4.5

Set of experiments detailing the LDA system accuracies for global and local features with all data provisions (Viola-Jones, 64x64 images, SMOTE, and extra samples).



DST	trial 1	trial 2	trial 3	trial 4	trial 5	avg
90	0.846496	0.846003	0.849106	0.87683	0.851324	0.853952
100	0.860893	0.860703	0.878692	0.864979	0.843575	0.861768
110	0.828644	0.853218	0.868142	0.869038	0.843042	0.852417
120	0.857624	0.880458	0.863391	0.863259	0.87451	0.867848
130	0.831513	0.877364	0.842304	0.867043	0.833864	0.850417
140	0.872153	0.866115	0.857027	0.863712	0.838898	0.859581
150	0.84706	0.867761	0.875134	0.850162	0.822675	0.852558

To lower the amount of time spent on experiments, only DST feature vectors from length 90 to 150 were evaluated. The results are in the Table above. The typical pattern previously seen with frequency-based features and their accuracy relative to vector size was not present. Instead, the accuracies went up and down as the vector length increases. They do not go below 85% but there is a noticeable difference between global-only systems and global + local systems.

In the end, using both global and local features performed roughly the same at the optimal lengths (120-140) when compared to purely global features. There was a 2% increase in the former compared to the latter. It is important to reiterate that the system accuracy hides important information, so getting the complete pictures requires examining the class accuracies. These will be discussed in Section 4.2.9 Best Performing Features using Cohn-Kanade.

Case for 20-Point Method. As previously detailed, the XPoint method was a novel approach to emotion classification. An experiment with this feature type consisted of passing a certain configuration of points and lines through a classifier (kNN), including data-fixing measures such as full SMOTE and extra samples. There was no need for pre-processing on the images because the features came out of geometric properties of the face rather than pixel information from the images.

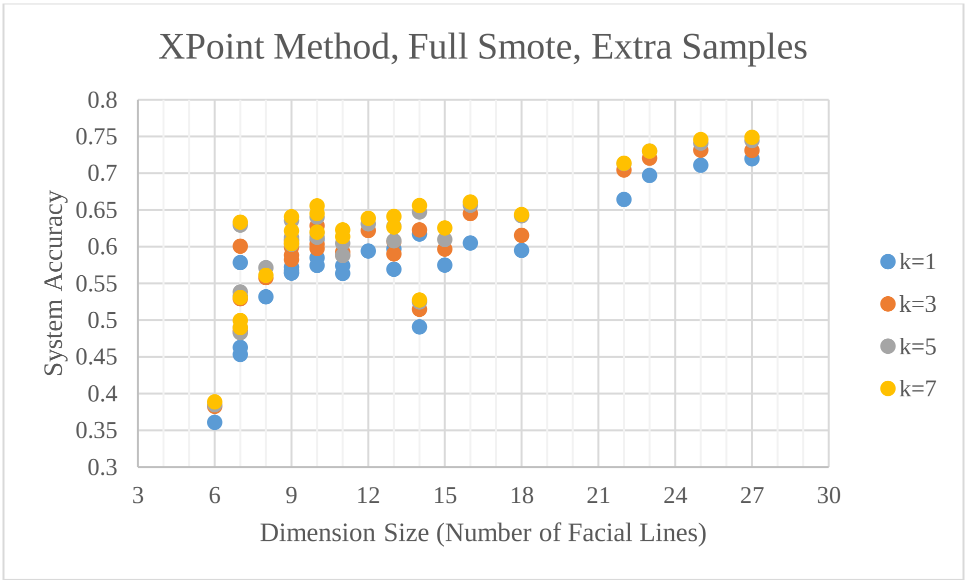


Figure 4.13. Set of experiments detailing the results of different XPoint features in a kNN Classifier.

kNN produced accuracies that were higher than LDA for XPoint features. The reason that there are multiple instances for some dimension sizes is because different sets of points produced the same number of lines. The general trend in Figure 4.13 shows that more dimensions meant greater accuracy. At the higher dimensions, however, the accuracy began to plateau. Further dimensions were not added after 27 for two reasons: there would likely be no appreciable change and there were no other imaginable configuration left for different point and line layouts.

The most accurate result (with 20 points and 27 lines) was 74.5% accurate, which was lower than system accuracies in previous successful experiments. Considering that XPoint was an exploration into simplifying FACS, the less accurate results were expected. The dimension sizes may have not been big enough, and many dimensions did not contain great differences amongst emotions. This was most apparent with subtler emotions such as sadness, which was hard to distinguish from neutral in some cases. Anger and disgust were similar in that respect as well. Moreover, the normalization line may have fixed scaling issues for the XPoint method, but if a person's face was sufficiently disproportioned, the lines were not reflective of their actual facial expression. The next section details the class averages from the most successful experiments so far and discusses them.

Best Performing Features Using Cohn-Kanade. The overall system accuracies from experiments provided a quick way to compare different feature extraction methods. However, they do not provide insight on how well the separate emotions were classified in that system. Perhaps one emotion was greatly under-performing but it's accuracy was averaged out by other high performing emotions. In this section, the three most distinct and best performing features sets were examined by comparing their class accuracies.

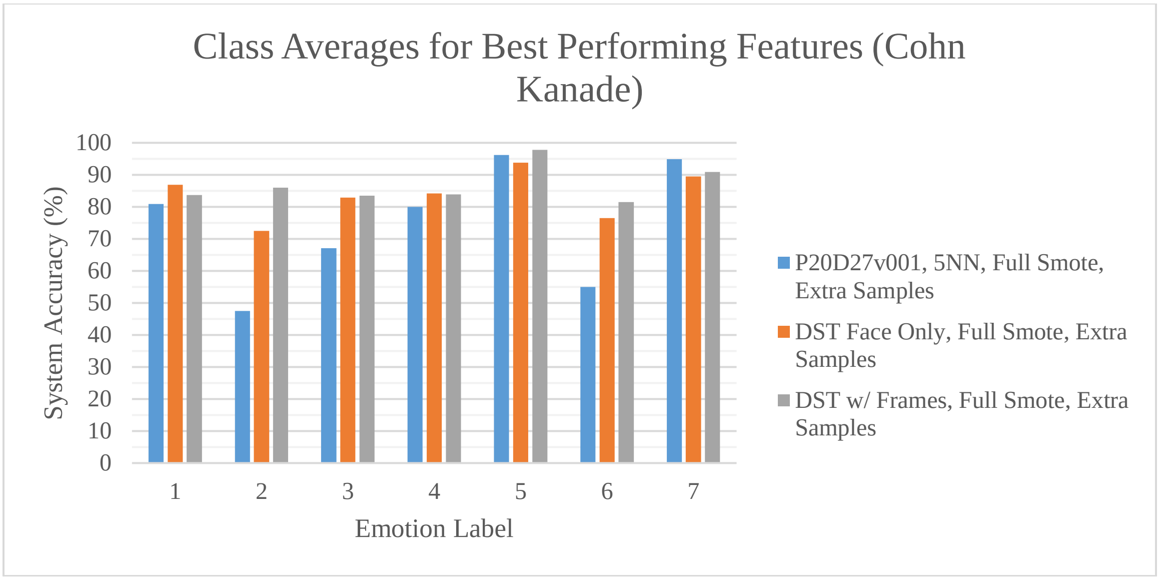


Figure 4.14. Class averages for best performing features from the Cohn-Kanade dataset.

Table 4.6

System accuracies for best feature extracted from Cohn-Kanade database.

Features	System Accuracy
P20D27v001	74.51153
DST, SMOTE, Extra Samples	83.76034
DST, SMOTE, Extra Samples, w/ Frames	86.76091

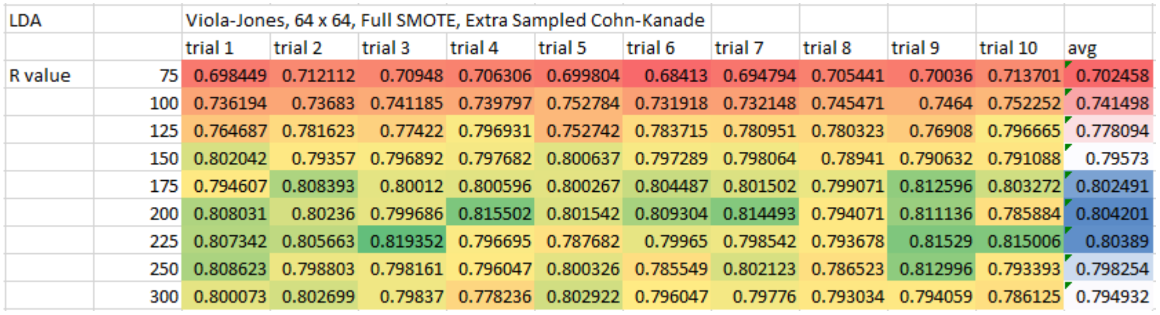
As expected, the fusion of global and local features (DST with Frames) yielded better results than using the face only. At the cost of a few percentage points from the neutral class, all other emotions performed better with frames. Anger saw the highest boost in accuracy, and sadness benefitted as well. In those instances, frames around the mouth gave the biggest clues for the emotion: the tightened lips of anger and the wrinkles at the corner of the pulled lips of sadness. When only the face was examined, these emotions would often be misclassified as neutral expressions.

The P20D27v001 configuration for the XPoint method did not perform up to expectation. The one surprising result was its comparable performance for fear. The other difficult expression (anger, disgust, and sadness) underperformed in comparison to the frequency-based features. This was most likely due to not enough discriminant dimensions to distinguish them from the neutral expression. The most obvious drawback to the method is its inability to account for face morphology. Some faces will have naturally bigger or smaller mouths, higher or lower eyebrows, and wider or narrower eyes. All of these factors may be interpreted as expressive features to the classifier when they are actually just part of a neutral pose.

PCA Results. Eigenfaces were another set of features that were tested to gauge their predictive capabilities when it came to classifying emotions on faces. Each face had a single PCA vector associated with it: a vector where each dimension detailed the amount that a corresponding mode of variation was present in the image. As with other experiments, these features vectors were placed in both a LDA and kNN classifier to compare their results. The LDA results are evaluated first.

Table 4.7

Set of experiments detailing the results of PCA features and an LDA classifier.



LDA		Viola-Jones, 64 x 64, Full SMOTE, Extra Sampled Cohn-Kanade										
		trial 1	trial 2	trial 3	trial 4	trial 5	trial 6	trial 7	trial 8	trial 9	trial 10	avg
R value	75	0.698449	0.712112	0.70948	0.706306	0.699804	0.68413	0.694794	0.705441	0.70036	0.713701	0.702458
	100	0.736194	0.73683	0.741185	0.739797	0.752784	0.731918	0.732148	0.745471	0.7464	0.752252	0.741498
	125	0.764687	0.781623	0.77422	0.796931	0.752742	0.783715	0.780951	0.780323	0.76908	0.796665	0.778094
	150	0.802042	0.79357	0.796892	0.797682	0.800637	0.797289	0.798064	0.78941	0.790632	0.791088	0.79573
	175	0.794607	0.808393	0.80012	0.800596	0.800267	0.804487	0.801502	0.799071	0.812596	0.803272	0.802491
	200	0.808031	0.80236	0.799686	0.815502	0.801542	0.809304	0.814493	0.794071	0.811136	0.785884	0.804201
	225	0.807342	0.805663	0.819352	0.796695	0.787682	0.79965	0.798542	0.793678	0.81529	0.815006	0.80389
	250	0.808623	0.798803	0.798161	0.796047	0.800326	0.785549	0.802123	0.786523	0.812996	0.793393	0.798254
	300	0.800073	0.802699	0.79837	0.778236	0.802922	0.796047	0.79776	0.793034	0.794059	0.786125	0.794932

Unlike the frequency-based features, the vector length had to be greater to produce better results. The optimal length was found through trial and error, as detailed above. A vector length around 200 dimensions showed the highest accuracies in these experiments, compared to the 120 optimum for DST. At this size, the accuracy topped out at 80%; this was 5% lower than high performance features in previous tests. At a length of 200, it would seem, the latter modes of variation are describing mostly noise in the image. However, the results show there is still valuable discriminatory information at such high modes for facial images.

The class accuracies with a vector length of 200 provided further discouragement for using PCA with LDA for future experiments. While all other class accuracies were above 70%, the neutral expression didn’t even perform better than random guessing (below 50%). It was most commonly misclassified as anger and sadness. This is most likely due to the nature of Eigenfaces. Since the difference in skin textures, like wrinkles, between subtle emotions like neutral, angry, and sad are low, the feature vectors for each of those emotions would not show great differences. PCA performed exceptionally well for bolder emotions like disgust, happiness, and surprise. These three emotions saw class accuracies above 95%.

Table 4.8

Confusion matrix for a PCA feature vector of length 200 in an LDA classifier.

	netural	angry	disgusted	fear	happy	sad	surprised
neutral	45.18359	17.42981	6.825054	6.933045	4.492441	14.42765	4.708423
angry	0.37037	73.7037	20.18519	2.592593	0	3.148148	0
disgusted	0	0.75	98.25	0	1	0	0
fear	0.487805	0.487805	2.439024	77.31707	17.31707	0	1.95122
happy	0	0	0.15625	0	98.90625	0	0.9375
sad	0.487805	6.585366	7.804878	2.926829	3.902439	75.12195	3.170732
surprised	0	0	0	0.526316	2.631579	0	96.84211

The kNN results for PCA mirrored those of frequency-based features. In Table 4.9, The accuracies did not surpass 30% for most feature vector lengths. This was true for shorter lengths not shown in the table, too. Once again, kNN was shown to not work well for highly dimensioned feature vectors.

Table 4.9

Set of experiments detailing the results of PCA features and a kNN classifier.

	trial 1	trial 2	trial 3	trial 4	trial 5	trial 6	trial 7	trial 8	trial 9	trial 10	avg
75	0.311605	0.339206	0.308554	0.296403	0.297698	0.306672	0.301066	0.284027	0.315689	0.321337	0.308226
100	0.302073	0.291042	0.31154	0.309737	0.33102	0.295293	0.304566	0.305172	0.303301	0.298626	0.305237
150	0.308964	0.292746	0.306074	0.288469	0.28103	0.291624	0.298645	0.295893	0.309227	0.297868	0.297054
200	0.294357	0.323531	0.30698	0.305067	0.29105	0.29009	0.313742	0.286079	0.284099	0.300587	0.299558

JAFFE Results

The JAFFE dataset had advantages and disadvantages when compared to Cohn-Kanade. The data distribution in terms of sample sizes for all of the emotions was much more even in JAFFE. There was no disparity amongst them. However, there weren't as many samples for each emotion. Compared to the extra-sampled Cohn-Kanade set, the least represented set (fear with 31 samples) is the same size as the most represented in JAFFE (fear and happiness with 31). SMOTE also had to be applied differently to the set, although the factors were easy to pick: each emotion was expanded 10-fold. This is reflected in Table 4.10 below.

Table 4.10

Emotion sample size distributions for the JAFFE database.

index	emotion	Viola-Jones sample size	SMOTE sample size
0	neutral	30	300
1	anger	30	300
2	contempt	29	290
3	disgust	32	320
4	fear	31	310
5	happiness	31	310
6	sadness	30	300
7	surprise	30	300

The JAFFE dataset also lacked a crucial component of information: landmarks. As previously stated, the Cohn-Kanade dataset had 68 points placed on key points of the face. These points allowed for XPoint and frame-based (local) features extraction. JAFFE, on the other hand, only consisted of facial images. A tool needed to be implemented that placed those 68 points on all JAFFE faces to progress with the same feature extraction techniques. The following section details the outcomes from designing, implementing, and executing that tool for images in the JAFFE database.

Automatic Facial Point Annotator. The accuracy analysis of the automated point annotator was conducted on the Cohn-Kanade database. The output from the tool on Cohn-Kanade faces was compared to the original points provided with the database to gauge how closely the tool placed points to their intended location. 100 faces from the database were chosen at random (with equal representation from all emotions) and used to assess the tool’s accuracy.

After the annotator placed the points on the 100 faces, the distance between those points and the actual landmark points from the database were examined. If a point placed by the tool was within a certain threshold distance (a few pixels in any direction), it was counted as a 100% correct placement. Otherwise, there was another larger region around the actual point that the placed point could reside in and still be counted as proportionally correct (going to 0% at the boundary of this region). The results were aggregated for four important regions of the face, seen in Table 4.11. The points that outlined the face were not counted since they were not used in any experiments.

Table 4.11

Accuracy of placed points from automated tool compared to Cohn-Kanade ground truth.

Regions	Accuracy
Brows	79.41
Eyes	94.21
Nose	91.64
Mouth	85.57

It is important to note that the accuracies would theoretically not remain as high when the tool was applied to the JAFFE images. Specifically, this is because the tool was trained on exclusively Cohn-Kanade images. As a result, the tool was trained to certain facial morphologies and lighting conditions. The textures and shapes varied between the two datasets because of racial and lighting differences. The point clouds were also less accustomed to the way that JAFFE subjects expressed anger: with a puffed cheek that distorted the mouth region in a way that was previously not observed in Cohn-Kanade. These factors had a demonstrably negative effect, apparent from the low accuracies yielded by XPoint experiments. Before this is discussed, the standard frequency-based experiments were run first and will be discussed first.

Comparisons to Cohn Kanade. Several experiments that were performed on the Cohn-Kanade database were mirrored in the JAFFE database to see if the results would be similar. If that was the case, then assumptions about certain parameters can be made to avoid re-running experiments in JAFFE. Without testing to see if it was necessary, the images were cropped via Viola-Jones and scaled to 64 by 64 pixels. Although the dataset was balanced in terms of sample sizes for each emotion, there was still room for SMOTE to be applied. Namely, the effects of enlarging each emotion sample size by 10 times was explored. The experiments started at this stage: comparing the effects of either applying SMOTE or not to the dataset.

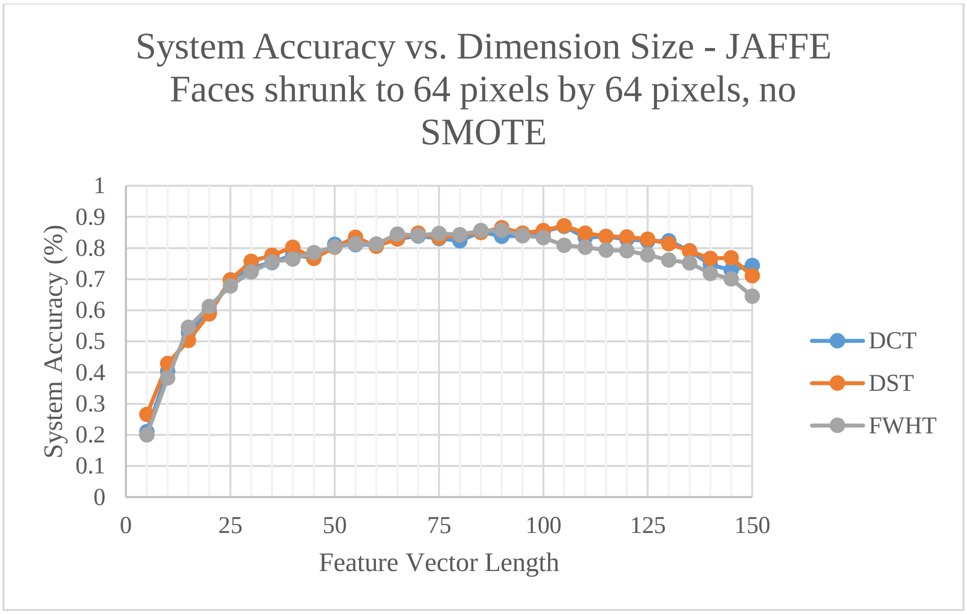


Figure 4.15. System accuracies for various feature extracted from JAFFE database, without using SMOTE.

Without SMOTE, a similar system accuracy curve (compared to Cohn-Kanade) was generated from the frequency feature experiments. Figure 4.15 details how higher dimensions generated higher accuracies up to a point. In fact, compared to similar Cohn-Kande experiments (Figure 4.4), the JAFFE results produced higher accuracies. This is the result of having a well-balanced dataset.

However, the highest accuracies on the curve appeared in a different range of dimensions for JAFFE. Rather than having an optimal dimension length for the features around 120, JAFFE frequency features performed better when they were around 100 dimensions in length and dropped shortly after instead of plateauing. While the direct reason why this was the case may be unknown, it was postulated that the way some emotions were posed may have had some effect. In the database, the poses performed by the subjects did not look visually similar for emotions like anger, disgust, and fear when compared to analogous American expressions. Moreover, the expressions in JAFFE were not presented as boldly as faces in the Cohn-Kanade set. More information per image (using higher dimensions) may have clouded the LDA classifier and made it less discriminant.

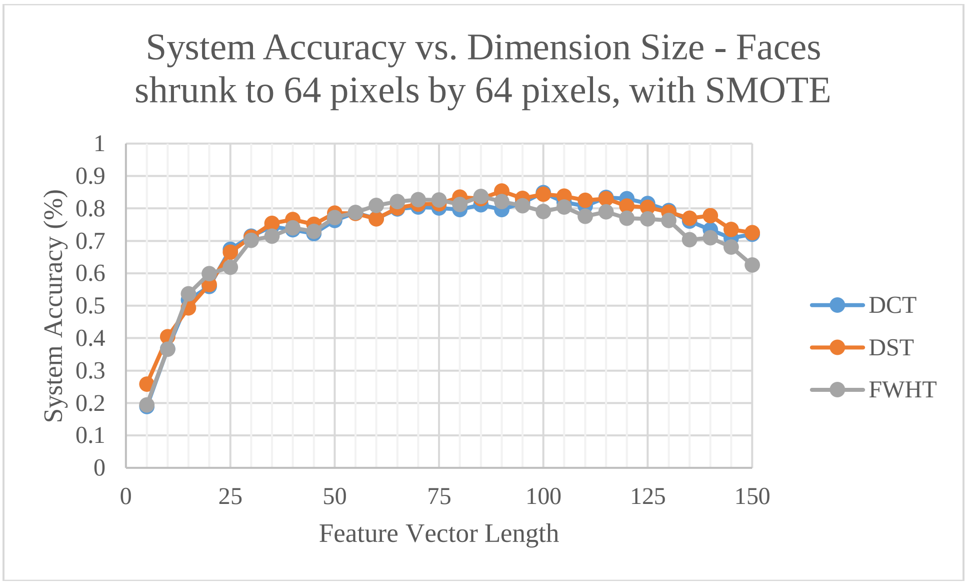


Figure 4.16. System accuracies for various feature extracted from JAFFE database, with SMOTE.

Although the data was already well balanced, SMOTE was applied to understand the repercussions. The sample sizes for each emotion were increased tenfold. The increase of synthetic data had very little impact, if not slightly negative. Figure 4.16 shows the same trend as experiments with no SMOTE, with lower accuracies. The hierarchy of accuracy also held true when compared to Cohn-Kanade results: DST performed better than DCT and DCT performed better than FWHT. In the end, it was confirmed that assumptions could be made for the JAFFE database that were proved to work on the Cohn-Kanade set. This included: using Viola-Jones, shrinking the image, and using frames for local feature extraction.

Best Performing Features Using JAFFE. As with the Cohn-Kanade database, the best performing features from those experiments were replicated on JAFFE images. The results can be seen in Figure 4.17 and Table 4.12 below. The hierarchy in terms of accuracy were the same: XPoint produced the least accurate system, followed by face-only DST, and DST with frames as most accurate. Overall, the latter two systems produced almost identical system accuracies when compared to Table 4.6. The only exception was the poor performance of XPoint on JAFFE images.

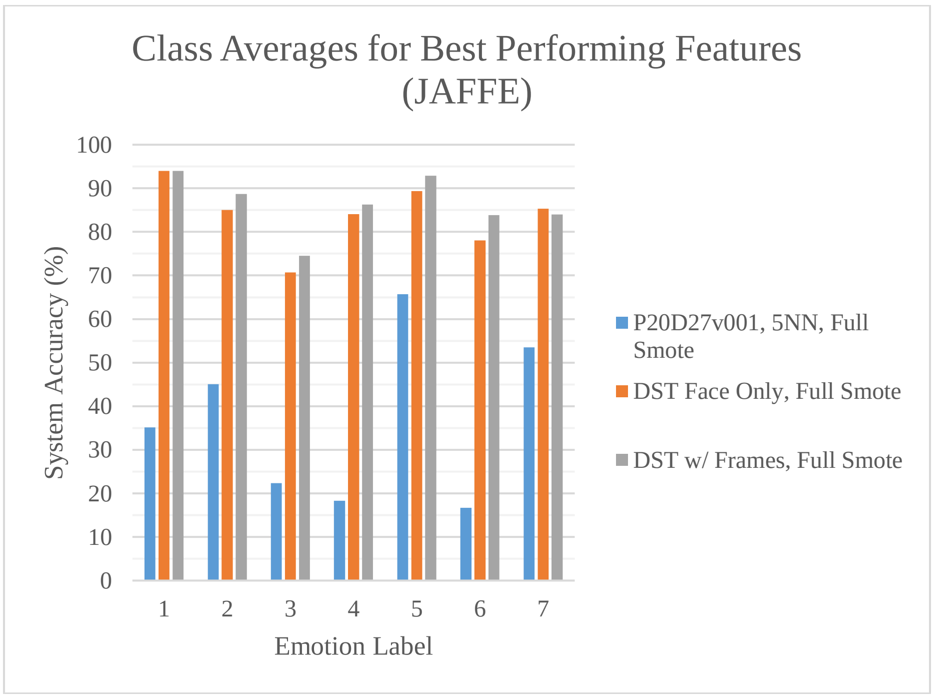


Figure 4.17. Class averages for best performing features from the JAFFE dataset.

Table 4.12

System accuracies for best feature extracted from JAFFE database.

Features	System Accuracies
P20D27v001	36.68713
DST, SMOTE, Face Only	83.78641
DST, SMOTE, w/ Frames	86.31052

There are two potential reason as to why P20D27v001 did not produce an accurate system. First, as previously stated, the images in the JAFFE database were did not feature posed emotions that were overly expressive. Many of them resembled a slight modification of the neutral pose. The 27 lines drawn from the 20 points may have been affected more by face morphology than facial expressions in that case.

The other more important issue is the point annotators accuracy. If the points were not correctly placed on JAFFE faces at the start of the experiments, then the performance of XPoint would suffer. The points placed on JAFFE images were visually inspected and it was concluded that, for the most part, the points were where they needed to be. However, there were a few images that had points partially or completely incorrectly placed. Since the sample sizes for each emotion were smaller than Cohn-Kanade sample sizes, those errors in placement would have a much stronger effect on XPoint accuracy. The tool cannot be completely at fault, since the training data came from a foreign dataset (Cohn-Kanade) which wasn’t highly representative of JAFFE images.

Fusion Results

The last set of experiments dealt with fusing multiple feature extraction methods into one system. There were four sets used in fusion tests: a fusion of purely frequency-based features (DST, DCT, and FWHT), a fusion of the previous three and the inclusion of DST frames, a fusion of DST, DCT, FWHT, and the P20D27v001 feature set, and a fusion of all of the above (DST, DCT, FWHT, DST Frames, and P20D27v001). In the case of Cohn-Kanade, each frequency feature had a dimension size of 120 (as it was found to be optimal in previous tests) and JAFFE utilized a dimension size of 95. Each feature had its own classifier but the training and testing data was synchronized across all of them. There was no fusion of databases in these experiments, and the details are separated in this section.

The first set of fusion experiments were on the Cohn-Kanade database images. Detailed in Figure 4.18 and Table 4.13, the results were not overly encouraging. There was a marginal increase in accuracy with the best performing fusion: all of the features produced a system that was 88% accurate. This was 2 percentage points higher than the best previous best performing feature set: DST with frames. The trend in Table 4.13 suggests that adding additional features to the base fusion (DST, DCT, and FWHT) can have a positive improvement on system accuracy. Even though the accuracy of P20D27v001 was under 75%, it still provided the system with a net benefit. The only area where fusions did not increase the accuracy of the system was with images portraying sadness. Whereas previous experiments had all class accuracies above 80%, none of the fusions could get sadness to perform better. The reason for this will be explained later.

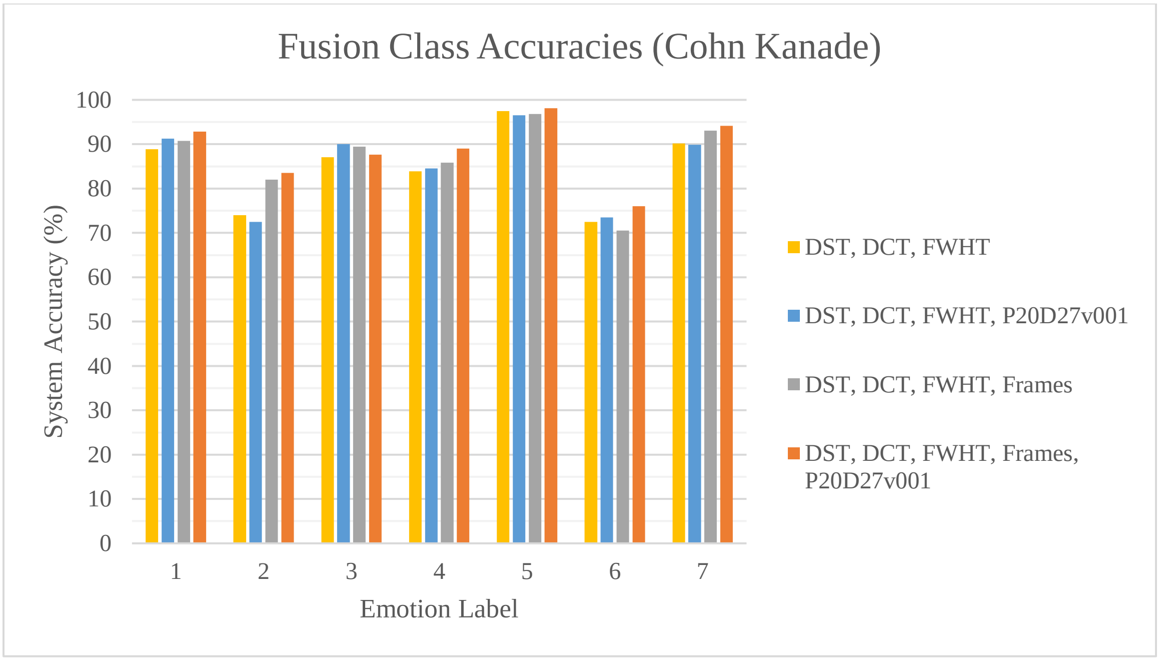


Figure 4.18. Class accuracies for Cohn-Kanade feature fusion experiments.

Table 4.13

System accuracies for Cohn-Kanade feature fusion experiments.

Features	System Accuracy
DST, DCT, FWHT	84.84599
DST, DCT, FWHT, P20D27v001	85.45215
DST, DCT, FWHT, Frames	86.90329
DST, DCT, FWHT, Frames, P20D27v001	88.74818

The JAFFE results did not mirror the Cohn-Kanade results. In fact, the fusion that contained all of the feature sets performed the worst (although the different fusion combinations mostly performed the same). In Table 4.14, the most accurate fusion was the most basic: the three frequency-based features. The explanation for this is mostly likely that the points placed on the face were not accurate enough. Recalling that P20D27v001 produced less than 40% accuracy with JAFFE images, it was noteworthy to see that fusion sets where P20D27v001 was present were not drastically impacted by the inaccuracy. Also, unlike the Cohn-Kanade results, the JAFFE fusions struggled more with the emotion of disgust than with sadness. This can be attributed to the way that disgust was posed in the JAFFE images, which did not differ greatly when compared to neutral expressions.

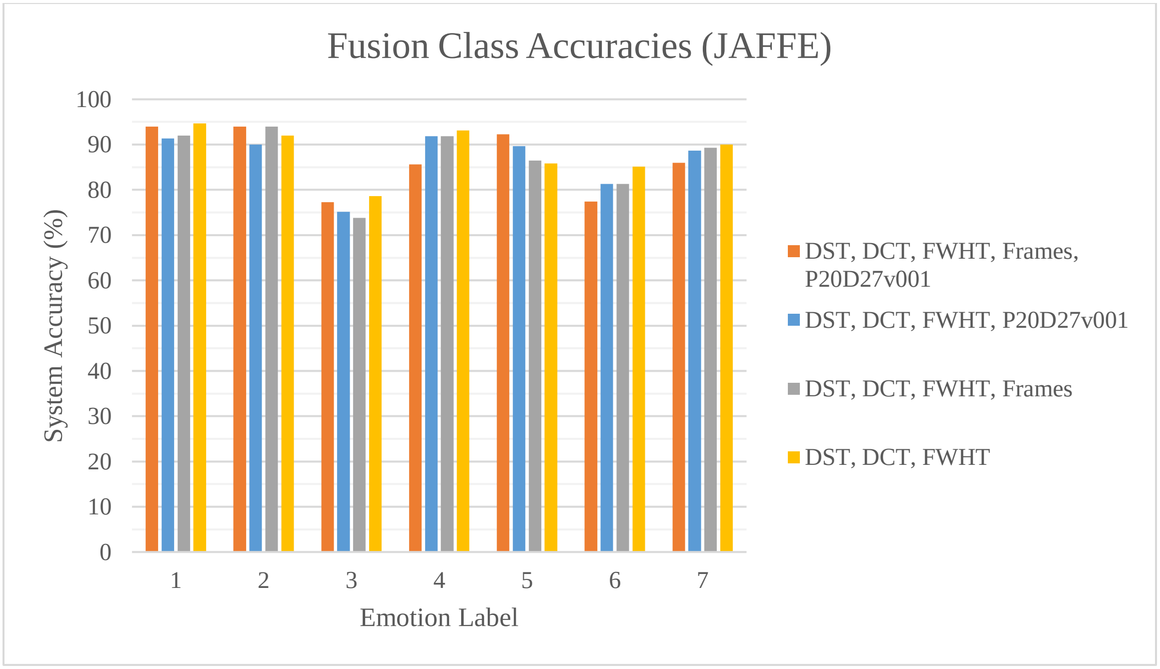


Figure 4.19. Class accuracies for JAFFE feature fusion experiments.

Table 4.14

System accuracies for JAFFE feature fusion experiments.

Features	System Accuracy
DST, DCT, FWHT, Frames, P20D27v001	86.64911
DST, DCT, FWHT, P20D27v001	86.85931
DST, DCT, FWHT, Frames	86.96334
DST, DCT, FWHT	88.48287

An important conclusion to draw from these fusion experiments is that the diversity of the features is important. The reason that sadness most likely could not break 80% accuracy in fusion tests was due to the similarity of the features. Most of them were frequency-based, and those types of features tended to misclassify sadness as neutral often. This misclassification was exacerbated by similarly acting feature sets, with only the chance of P20D27v001 offsetting the bias. For the same reason, the results were not impacted too much by P20D27v001 in JAFFE even though they were a detrimental feature given the point annotators faults. The redundant use of DST, DCT, and FWHT in the fusions pushed the systems to perform as if the same features was repeated three times.

The fusion experiments were helpful in increasing the overall system accuracy to its highest value in the research: 88%. While this did not break the 90% set up by the hypothesis, it came fairly close. This research shows the argument for feature fusion, as it has the ability to produce some double-checking measures for the classifier results and slightly fixes incorrect results. The fusion systems are not overly complicated or computationally demanding because they are a parallel arrangement of classifiers with most of the heavy computations happening offline in the training stage. The last chapter will take in all of the presented results and draw conclusions from them.

Chapter 5

Conclusion

Summary of Findings

Re-examining the hypotheses proposed at the beginning of this thesis helps to elucidate whether the research presented here produced meaningful results. The first prediction from the hypothesis was that using simple frequency-based features would lead to a reasonably accurate emotion classification system. While the results from these experiments did not yield accuracies over 90%, the system accuracies for both Cohn-Kanade and JAFFE images were above 80% when DST, DCT, and FWHT were applied to the face (without the use of frames). The mix of frequency-based features and an LDA classifier proved to be a promising place to start the investigation into facial emotion recognition systems.

The next prediction was that including frames (local features) would further raise the accuracy of a system. This point was demonstrably confirmed, as experiments showed that adding frames benefited the system as compared to a system that used the entire face as a single feature. The frames were extracted from feature-rich areas of the face, where contortions from different emotional expressions provided extra discriminant values that raised the accuracy of the system.

The third prediction, that a tool which automatically placed landmarks on a face could be made with a novel frequency-based search criterion, was partially confirmed. Specifically, the use of DST patches in the neighborhood of each point proved to be an effective means of pushing the point into the correct direction if it was out of place. The tool worked almost as well as the original algorithm, given that the training data was an accurate representation of the images that the tool was applied to.

The XPoint method, while novel, was not as accurate as other methods used in this research for facial emotion feature extraction. While the method is easy to understand, is computationally simple, and can be analyzed with a simple kNN classifier, the resulting system accuracy was around 75%. A weakness of XPoint included the relatively poor detection of subtler emotions, and the feature extraction method did not have a way to compensate for naturally variant facial morphologies (such as naturally raised brows or smaller/larger mouths). Although XPoint did not yield comparably high accuracies, the method was still useful for providing diversity in the results concerning the last prediction from the hypothesis: feature fusion would further increase a system's accuracy.

As the results have shown, when multiple features are fused to make a decision about a subject's facial emotional expression, the overall system accuracy increases. The last set of experiments generated accuracy results of 88%, close to the 90% level of a 'reasonably accurate' system. Systems that incorporated fusion were robust because multiple classifiers were available to compensate for discrepancies during their voting, even if one of their classifiers underperformed. As the thesis stands, most of the propositions have been proven true, and an accurate emotional classification system was created using knowledge of various feature extraction techniques.

Recommendations and Feasibility

The research presented here was intended to demonstrate the creation of an emotion recognition system through the employment of various feature extraction and classification techniques. Many of the choices made honored the intention of keeping algorithms computationally simple in order to preserve the opportunity for production-level deployment of the system. As there are many different specific applications for emotion recognition systems (as discussed in Section 1.3), these specific applications would also require different interfaces for the intended end-user. However, regardless of how the development of those interfaces is handled, this research aimed to provide the following: clear entry points into the system, explanations of the pre-processing, feature extraction, classification, and fusion, and the intended output. With those fundamental components, any other peripheral elements like cameras or graphical user interfaces would not be difficult to incorporate.

Future Work

Due to the expansive nature of the machine learning field, there are still many existing feature extraction techniques and classifiers that were not tested in this research. While the approaches presented in this thesis were comparatively more simplistic, these other existing methods bring into play mathematical principles that are much more complicated. Such methods include deep learning algorithms and neural network, which attempt to emulate the structure of the human brain in order to classify input data. Since the simple and effective solutions used in this research were demonstrated to be useful, future work may include the use of these other more complicated algorithms to further increase system accuracy.

There was some novelty in the approach to the Active Shape Model algorithm in this research, but future work may include examining Active Appearance Models as the main component for an automatic facial point annotator. Aside from potentially increased accuracy in placing points, these landmarks could also be used in a Facial Action Coding

System classification system. In such a system, the various microexpressions of a face could be classified as their correct Action Unit, and those AUs could then be classified as an emotion. Lastly, the use of local features during facial feature extraction offers some groundwork for solving the occlusion problem: faces that have objects that block or obscure parts from view.

There are many directions to take this research into, and the work in this thesis has shown that even solutions which are relatively simple can produce satisfactorily accurate results. The future for emotion recognition systems may be expansive, but it is also bright. Technological advances help man and machine have a symbiotic relationship, and the further comprehension of emotion from artificially intelligent partners offer even greater value to the human race.

References

Bhadu, A., Tokas, R., & Kumar, V. (2012). Facial Expression Recognition Using DCT, Gabor, and Wavelet Feature Extraction Techniques. *International Journal of Engineering and Innovative Technology*, 92-95.

Birdwhistell, R. L. (1952). *Introduction to Kinesics: An annotation system for analysis of body motion and gesture.* Ann Arbor: University of Michigan Library.

Bunke, H., & Ha, T. M. (1997). Handwritten Numeral Recognition by Perturbation Method. *Pattern Analysis and Machine Intelligence*, 76.

Busso, C., Deng, Z., Yildirim, S., Bulut, M., Lee, C. M., Kazemzadeh, A., . . . Narayanan, S. (2004). Analysis of Emotion Recognition using Facial Expressions, Speech and Multimodal Information. *Proceedings of the 6th international conference on Multimodal interfaces*, 205-211.

Butalia, A., Ingle, M., & Kulkarni, P. (2012). Facial Expression Recognition for Security. *International Journal of Modern Engineering Research*, 1449-1453.

Calder, A. J., Burton, A. M., Miller, P., Young, A. W., & Akamatsu, S. (2001). A principle component analysis of facial expressions. *Vision Research*, 1179-1208.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research 16*, 321-357.

Cootes, T. F., Edwards, G. J., & Taylor, C. J. (1999). Comparing Active Shape Models with Active Appearance Models. *British Machine Vision Conference* (pp. 173-182). Nottingham: CRI Repro Systems Ltd.

Cootes, T. F., Taylor, C. J., Cooper, D. H., & Graham, J. (1995). Active Shape Models - Their Training and Application. *Computer Vision and Image Understanding*, 38-59.

Darwin, C. (1899). *The Expression of Emotion in Man and Animals.* New York: D. Appleton And Company.

Discriminant Analysis. (2015, December 30). Retrieved from Mathworks: http://www.mathworks.com/help/stats/discriminant-analysis.html#bs31mtt

Ekman, P. (1971). Universals and Cultural Differences in Facial Expressions of Emotion. *Nebraska Synposium on Motivation* (pp. 207-282). Lincoln: University of Nebraska Press.

Ekman, P. (1998). Afterword: Universality of Emotional Expression? A Personal History of the Dispute. In C. Darwin, *The Expression of the Emotions in Man and Animals* (pp. 363-393). New York: Oxford University Press.

Ekman, P. (1999). Basic Emotions. In *Handbook of Cognition and Emotion* (pp. 45-60). New York: John Wiley & Sons Ltd.

Ekman, P., Friesen, W. V., & Tomkins, S. S. (1971). Facial Affect Scoring Technique: A First Validity Study. *Semiotica*, 37-58.

Facial Action Coding System. (2016, January 8). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Facial_Action_Coding_System

Faundez-Zanuy, M., & Fabregas, J. (2007). On the Relevance of Facial Expressions for Biometric Recognition. *Verbal and Nonverbal Features of Human-Human and Human-Machine Interaction* (pp. 32-43). Patras: COST Action 2102 International Conference.

Guney, F. (n.d.). *Emotion Recognition using Face Images.* Istanbul: Bogazici University.

Hadamard Transform. (2016, March 1). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Hadamard_transform

Hassan, M., Osman, I., & Yahia, M. (2007). Walsh-Hadamard Transform for Facial Feature Extraction in Face Recognition. *Proceedings of World Academy of Science, Engineering and Technology*, 194-198.

Jack, R. E., Garrod, O. G., & Schyns, P. G. (2014). Dynamic Facial Expressions of Emotion Transmit an Evolving Hierarchy of Signals over Time. *Current Biology*, 187-192.

Losh, M., & Capps, L. (2006). Understanding of Emotional Experience in Autism: Insights From the Personal Accounts of High-Functioning Children With Autism. *Developmental Psychology*, 809-818.

Lucey, P., Cohn, J. F., Kanade, T., Saragih, J., & Ambadar, Z. (2010). *The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression.* Pitssburgh: Disney Research.

Matsumoto, D. (1991). Cultural Influences on Facial Expressions of Emotion. *The Southern Communication Journal*, 128-137.

Merriam-Webster. (2015, February 10). *Emotion*. Retrieved from Merriam Webster Online: http://www.merriam-webster.com/dictionary/emotion

Microexpressions. (2016, March 13). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Microexpression#cite_ref-ekman_1999_4-0

News Editor, P. (2015, November 25). *Facial Expressions Are Innate, Not Learned*. Retrieved from Psych Central: http://psychcentral.com/news/2008/12/30/facial-expressions-are-innate-not-learned/3570.html

Paithane, A. N., Hullyalkar, S., Behera, G., Sonakul, N., & Manmode, A. (2014). Facial Emotion Detection using Eigenfaces. *International Journal of Engineering and Computer Science*, 5714-5716.

Principle Component Analysis. (2015, November 25). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Principal_component_analysis

Saneiro, M., Santos, O. C., Salmeron-Majadas, S., & Boticario, J. G. (2014). Towards Emotion Detection in Educational Scenarios from Facial Expressions and Body Movements through Multimodal Approaches. *The Scientific World Journal*, 14 Pages.

Shergill, G. S., Diegel, O., Sarrafzadeh, A., & Shekar, A. (2008). Computerized Sales Assistants: The Application Of Computer Technology To Measure Consumer Interest - A Conceptual Framework. *Journal of Electronic Commerce Research*, 176-191.

Smirnov, D. V., Muraleedharan, R., & Ramachandran, R. R. (2015). A Comparison of Facial Features and Fusion Methods for Emotion Recognition. *International Conference on Neural Information Processing* (pp. 574-582). Instanbul: Springer International Publishing.

Villagrasa, S., & Susin, A. (2009). FACe! 3D Facial Animation System based on FACS. *IV Iberoamerican Symposium in Computer Graphics* (pp. 201-207). Margarita Island: Sociedad Venezolana de Computacion Grafica.

Viola, P., & Jones, M. J. (2004). Robust Real-Time Face Detection. *International Journal of Computer Vision*, 137-154.

Viola-Jones object detection framework. (2015, December 20). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework

Wang, S., Liu, Z., Lv, S., Lv, Y., Wu, G., Peng, P., . . . Wang, X. (2010). A Natural Visible and Infrared Facial Expression Database for Expression Recognition and Emotion Inference. *IEEE Transactions of Multimedia*, 682-691.

Zheng, W. (2014). Mulit-View Facial Expression Recognition Based on Group Sparse Reduction-Rank Regression. *IEEE Transactions on Affective Computing*, 71-85.