# ABSTRACT

A fruitful way for creating any innovative system in a computing environment is to integrate a sufficiently user-friendly interface with the ordinary end user. Successful design of such a user-friendly interface, however, means more than just the ergonomics of the panels and displays. It also requires that engineers precisely define what information to use and how, where, and when to use it. Facial expression as a natural, non-intrusive and efficient way of communication has been considered as one of the possible inputs of such interfaces. The research interest in facial expression recognition has grown due to its potential applications. Many local feature representations such as Gabor filters, LBP had been proposed for facial expression recognition. However, accuracy rates and running time of facial expression recognition achieved by these representations have yet to be improved. The work of this thesis aims at designing a robust and effective Facial Expression Recognition (FER) system that uses simple but effective gray scale invariant local descriptors for face expression recognition. The local feature pattern at a pixel is computed based on differences of gray color values of its neighboring pixels. The pattern represents the changes of gray color values of pixels in its surrounding area. Four alternative local feature representations are proposed depending on the numbers of considered directions and neighboring pixels. To create the feature vector for an image, the facial image is divided into blocks and the histograms counting the occurrences of all possible local patterns for all blocks are computed then concatenated. A variance-based feature selection method is also proposed to reduce the length of the descriptor, thus, help the running time for feature extraction, training and classification time. It can also be shown that the more the numbers of considered directions and neighboring pixels, the richer the information the local pattern represents and so the higher accuracy of facial expression recognition achieved from using the local pattern. However, longer pattern length and longer processing time will be needed. Experimental results show that the proposed feature representations along with Support Vector Machine are more effective than some other well-known local feature representations for facial expression recognition. The feature representations are also easy to compute and suitable for real time applications.

# ACKNOWLEDGEMENTS

More than anyone in this world, I would like to thank my Mom and Dad for their endless sacrifices and supports for making it possible to embark on this journey. To have such lovely parents in my life is truly an honor and a blessing. I hope to lead a professional and personal life that will make them proud.

I owe much gratitude to Associate Professor Dr. Surapong Auwatanamongkol, my PhD advisor, mentor and friend. He is someone you will instantly love and never forget once you meet him. I hope that I could be as lively, enthusiastic, and energetic as him and to someday be able to guide naughty researcher as well as he can. His time, efforts and insightful discussions throughout this degree are highly appreciable. I am also grateful to my committee members: Associate Professor Dr. Pipat Hiranvanichakorn, Assistant Professor Dr. Ohm Sornil and Assistant Professor Dr. Rawiwan Tenissara for their helpful comments and suggestions.

I am thankful to National Institute of Development Administration for providing me the full scholarship throughout this degree period.

Finally, I would like to thank my two younger brothers Babu and Shuvo, my wonderful wife, Shati, and beautiful daughter, Sneha. Thanks for supporting me during my studies and urging me on. Mom and Dad, you are wonderful parents and wonderful friends. Babu and Shuvo, I could not ask for better brothers and friends. Shati, my wife, if I wrote down everything I ever wanted in a wife I would not have believed I could meet someone better! And Sneha, my daughter, you have booked a year from my life by saving it from this degree. Your charming look and innocent questions had worked as a tonic throughout my study. Thanks my little angel for helping me finish this degree at its earliest time possible. It's about time! Remember, it's o.k. to stress just not to stress out. To all of you, thanks for always being there for me.

Mohammad Shahidul Islam

July 2013

# TABLE OF CONTENTS

# LIST OF TABLES

**Tables** **Page**

# LIST OF FIGURES

# CHAPTER 1

# INTRODUCTION AND BACKGROUND

## 1.1 Introduction

Facial Expression plays an important role in human-to-human interaction, allowing people to express themselves beyond the verbal world and understand each other from various modes. Some expressions incite human actions, and others fertilize the meaning of human communication. (Mehrabian, 1968: 53-55) mentioned in his paper that the verbal part of human communication contributes only 7%, the vocal part contributes 38% and facial movement and expression gives 55% to the meaning of the communication. This means that the facial part does the major contribution in human communication.

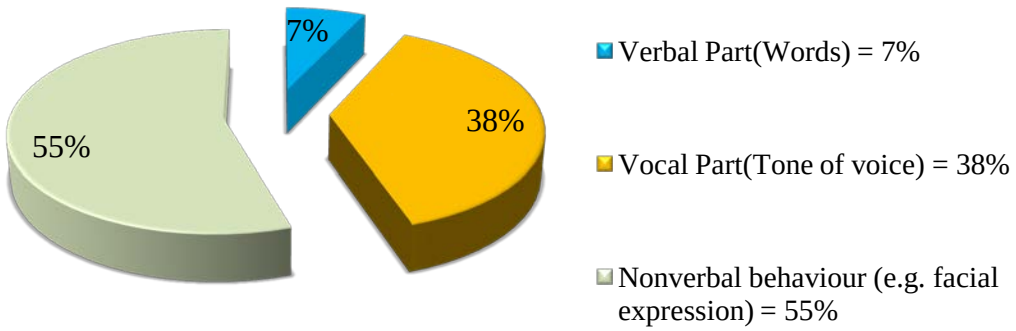



**Figure 1.1** 7-38-55 Rule by Mehrabian

Due to its potential important applications in man-machine interactions, automatic facial expression recognition has become a challenging problem in computer vision and has attracted much attention of the researchers in this field (Zeng et al., 2009: 39-58).

Some important applications of FER are,

1) Video surveillance for security,
2) Driver state monitoring for automotive safety,
3) Educational intelligent tutoring system (ITS),
4) Clinical psychology, psychiatry and neurology,
5) Pain assessment,
6) Image and video database management and searching,
7) Lie detection and so on

FER can play vital role in many areas of research and applications. “How humans recognize their emotions and use them to communicate information” is an important issue in Anthropology. Automatic facial expression or emotion recognition by man-made machines can be used in clinical psychology, psychiatry and neurology. Expression recognition can be embedded into a face recognition system to improve its robustness. e.g. in a real-time face recognition system where a series of images of an individual are captured, FER module picks the one which is most similar to a neutral expression for recognition, because normally a face recognition system is trained using neutral expression images. In Human Computer Interface (HCI), expression is a great potential input. This is especially true in voice-activated control systems. As mentioned by (Mehrabian, 1968: 53-55) when people are speaking, 55% of communication happens via expression whereas only 7% happens via spoken words. This implies that a FER module can markedly modify the performance of such systems. Companies and Service Providers can also gather customers’ facial expressions as implicit user feedbacks to improve their services in the future. Compared to a conventional questionnaire-based procedure, this is not only reliable but also virtually cost effective. Facial expression estimated from real images can be used to animate synthetic characters (Choi and Kim, 2005: 907-914). This is useful in video telephony where instead of transmitting the high bandwidth video of facial images, one can just send the facial expression sequence and the original video can be reconstructed from the sequence. This technique has also been used in the animation movie industry where high quality computer animation can be created from the facial expression sequence.

In general, a facial expression recognition system can be divided into three

modules, i.e. face acquisition, facial data extraction and representation, and finally facial expression recognition.

### 1.1.1 Face Acquisition

Facial images from different databases have diverse formats, resolutions, backgrounds and are taken under varying illumination. In general, face acquisition is also called image preprocessing and has two major subsections namely face detection and face alignment. Face detection finds locations and dimensions of human faces in digital images. Many algorithms implement the face detection task as a binary pattern classification. A given image is transformed into features part by part, after which a classifier trained on example faces decides whether the given part of the image is a face, or not. Sliding window is another popular technique for face detection. It means the classifier is used to classify the (usually square or rectangular) parts of an image, at all locations and scales, as either face or not a face (background pattern). Colors of skin can also be used to find face segments but it is a weak technique. Some database may not have all the necessary skin colors. Results can be also affected by lighting. The advantages are less restricted to orientation or size of faces and a good algorithm can work better with complex backgrounds (Colmenarez, Frey and Huang, 1999: 592-597). Faces are usually moving objects in real time videos. Separating the moving area can give the face segment. But, the video may have other moving objects as well which can affect the results. Blinking is a specific type of motions on faces. Determining a blinking object pattern in a video or image sequence can ensure the presence of a face (Reignier, 1995). Both the eyes usually blink together and their positions are almost symmetrical. Each image content can be subtracted from the previous image content in the video image sequence. The differences between the two image contents will show boundaries of the moved pixels. If the eyes happen to be blinking, there will be a small boundary within the face. Various sliding window shapes such as oval, rectangle, round, square, heart, and triangle, can be circulated over an image for face tracking. Once the face region is determined, the model is laid over the face and the system is able to track that face region movements. Another method for human face detection from color images or videos is to combine methods of image segmentation using colors, shapes, and textures. Using a skin color model,

objects of the skin color can be extracted from an image. Next, face models can be applied to the objects to eliminate false detections from the skin color model and to extract facial features such as eyes, nose, and mouth.

Face alignment refers to aligning one face with respect to another or a referenced line. It is also known as face registration. It can be done using either appearance based registration or feature based registration. Two popular algorithms for face registration are Active Appearance Model (AAM) and Constrained Local Model (CLM). An active appearance model (AAM) is an algorithm for matching a given statistical model of an object shape onto a new image. The statistical model is built during a training phase. A set of images, together with coordinates of manual landmarks that appear in all of the images, is used to build the model in the training phase. Cootes et al. introduced this model in (Cootes, Edwards and Taylor, 2001: 681-685). The algorithm is widely used for registering faces. The Constrained Local Model (CLM) (Cristinacce and Cootes, 2008: 3054-3067) algorithm is more robust and more accurate than the AAM, which relies on the image reconstruction error to update the model parameters. It aims to build a generic model of a class of objects, so that the model can fit to any new instance of the objects automatically. The CLM is efficient and robust method for locating a set of feature points in an object of interest.

### 1.1.2 Facial Data Extraction and Representation

Facial feature extraction is the process of converting a face image into a feature vector carrying characteristics of the face. The vector is used as the basis for expression class differentiation. Feature extraction is the vital part of most pattern-recognition tasks. Three types of feature extraction techniques commonly used in existing expression recognition systems are feature-based method, appearance-based method and hybrid method.

The feature-based method uses geometric features like facial points or shapes of facial components or spatial locations e.g. FACS. The appearance-based method uses colors, color layouts or textures of the facial skin including wrinkles and furrows e.g. local feature based methods, and the hybrid method uses both geometric and appearance facial features. All these visual features can be extracted either from the entire image or from regions. Global feature or feature for the whole image is

relatively simpler, while region based representation of images is proved to be more consistent to humans' perception. Some techniques are explained below:

1.1.2.1 Color Feature (appearance-based)

Most widely used features in image processing are the color features. Colors are defined on some certain color spaces, some commonly used ones include RGB, LAB, LUV, HSV and YCrCb. The main advantages of the color features are that it is comparatively robust to the background hazards and independent of image dimension and orientation. Color features can be holistic or region based. Color histogram is the most commonly used color feature. Along with color histogram, color moments, color sets and some other color representations are also used as features. Color moments are proposed by (Stricker and Orengo, 1995: 381-392) to overcome the quantization effects in the color histogram. To alleviate fast search over large-scale digital image collections, (Smith and Chang, 1996: 426-437) suggested color sets as an approximation to the color histogram.

1.1.2.2 Texture (appearance-based)

Image texture is the measurement of the spatial variation of grey tone values. It refers to the visual pattern results from different color or intensity. It gives important structural information which is important to differentiate many real-world images such as fruit skin, fabric, trees and clouds. These features usually used to draw the visual information including spectral features and statistical features characterizing texture by using local statistical measures. Coarseness, directionality, regularity, contrast, line-likeness, contrast and roughness are the tamura-features proposed by (Tamura, Mori and Yamawaki, 1978: 460-473). Among these six features, the first three are more important for texture description than the other three. Among the various texture features, Gabor texture and wavelet texture are widely used for image processing and have been reported to nearly match the perception of human vision. Texture can also be either holistic or local based feature.

1.1.2.3 Shape (geometric-based)

Some objects or content-based visual information retrieval applications need the shape representation of the object. The shape of the object should be invariant to translation, rotation and scaling. These specific features include aspect ratio, circularity, Fourier descriptors, moment invariants, consecutive boundary

segments, etc.

1.1.2.4 Spatial location (geometric-based)

Spatial location is also useful for region-based retrieval like color and texture description for facial or content base image retrieval. For an example, ‘sky’ and ‘sea’ could have almost similar color and texture features. But their location or spatial position on the image is different. The sky usually appears at the top of an image, while sea at the bottom. The spatial location can be specified in two ways: (a) absolute spatial location such as upper, bottom, top, and center, or (b) relative spatial relationship, such as the directional relationships between objects: left, right, above or below.

1.1.2.5 Handling layout (hybrid) information

Global features e.g. colors, textures, edges are simple to calculate and can give reasonable discriminating power in visual information processing but they tend to give too many false positives in case of large-scale image database. Many investigation results suggested that using local features along with spatial relations is a better method for image processing (Smith and Chang, 1996: 426-437). To change the global feature to a local one, a natural approach is to divide the image into sub-images and extract features from each of the sub-images. One way of this approach is the quadtree-based approach, where the entire image is split into a quadtree structure and each branch had its own characteristic to describe its content.

1.1.2.6 Facial Action Coding (geometric-based)

The basic prototypes of facial expressions are neutral, contempt, fear, sadness, disgust, anger, surprise and happiness (Ekman, 2005: 45-60). Most of the facial expression recognition systems (FERS) are based on the Facial Action Coding System (FACS) (Pantic and Rothkrantz, 2000: 1424–1445; Tian, Kanade and Cohn, 2001: 97–115; Tong, Liao and Ji, 2007: 1683–1699), originally developed by (Ekman and Friesen, 1978). Ekman, Friesen and Hager published a significant update to FACS in 2002. FACS (Hamm et al., 2011: 237-256) encodes movements of individual facial muscles (Figure 1.2). Categorized physical expressions of emotions are more convenient to psychologists and to animators. FACS works as a computed automated system that detects faces in videos, extracts the geometrical features of the faces. Using these features, it develops temporal profiles of each facial movement.

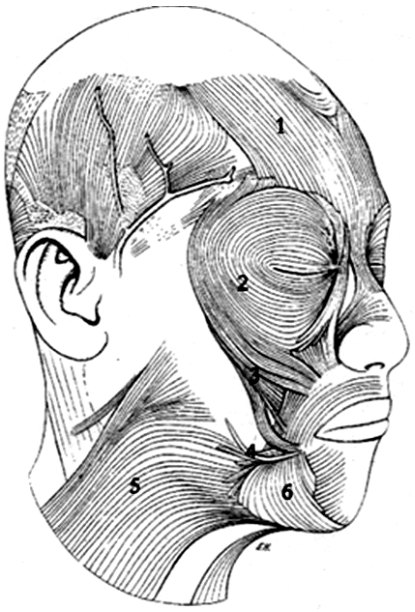



**Figure 1.2** Muscles of Facial Expression, 1) Frontalis; 2) Orbicularis oculi; 3) Zygomaticus major; 4) Risorius; 5) Platysma; 6) Depressor Anguli Oris

**Source**: Huber, 1931.

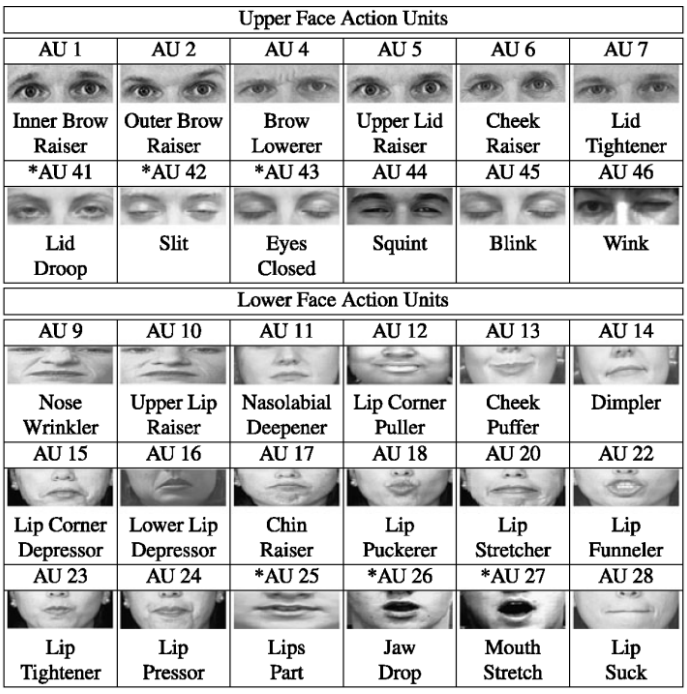



**Figure 1.3** Example of some FACS action units (AU)

**Source:** Tian et al., 2011: 490.

FACS is coded using 44 different action units (AUs), each of which is related to the facial muscle movements (Figure 1.3). The 44 action units can give up-to 7000 different combinations, with wide variations due to age, body shape and ethnicity. Some combinations of the AUs are shown in Figure 1.4.

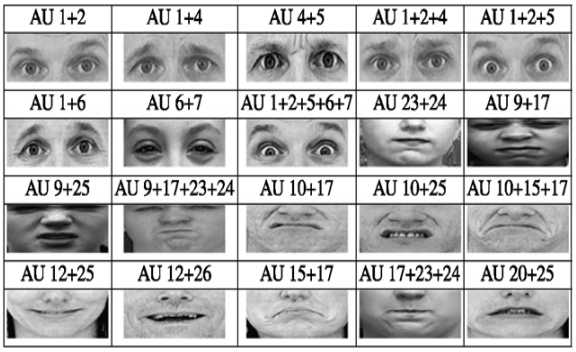



**Figure 1.4** Some Examples of Combination of FACS Action Units
**Source:** Tian et al., 2011: 491.

EMFACS (Emotional Facial Action Coding System) by (Friesen and Ekman, 1983) and FACSAID (Facial Action Coding System Affect Interpretation Dictionary**)** by (Ekman, Rosenberg and Hager, 1998) consider only emotion-related facial actions. Examples of these are:

**Table 1.1** Combination of AU's Indicating Specific Facial Expression

| Emotion | Action Units |
|---|---|
| Anger | 4+5+7+23 |
| Contempt | R12A+R14A |
| Disgust | 9+15+16 |
| Fear | 1+2+4+5+20+26 |
| Happiness | 6+12 |
| Sadness | 1+4+15 |
| Surprise | 1+2+5B+26 |

**Source:** Ekman et al., 1998.

1.1.2.7 Local Feature Representation (appearance-based)

Several local feature representations had been proposed for facial expression recognition. The local features are much easier for extraction than those of AUs. (Ahonen, Hadid and Pietikäinen, 2006: 2037-2041) proposed a new facial representation strategy for still images based on Local Binary Pattern (LBP). The method is basically proposed by (Ojala, Pietikäinen and Harwood, 1996) for texture analysis. In this method, the LBP value at the center pixel of a 3x3 region is computed using the gray scale color values of that pixel and its neighboring pixels as follows:

$$\text{LBP} = \sum_{i=1}^{\text{P}} 2^{i-1} f(\text{a(i)} - \text{e}) \tag{1.1}$$

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

Where e denotes the gray color value of the center pixel, a(i) is the gray color value of its neighbors, P stands for the number of neighbors, i.e. 8. Figure 1.5 shows an example of obtaining LBP value of the center pixel for a given 3×3 pixels region.

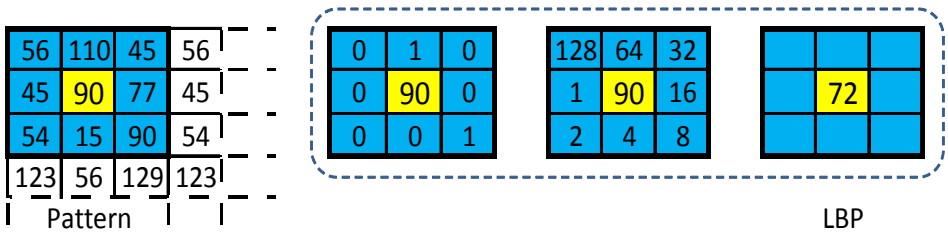



**Figure 1.5** Example of Obtaining LBP for a 3x3 Pixels Region/Area

An extension to the original LBP operator called uniform and rotation invariant local binary pattern ($\text{LBP}_{\text{RIU2}}$) was proposed by (Ojala and Pietikäinen, 2002). It can reduce the length of the feature vector and implement a simple rotation-invariant descriptor. An LBP pattern is called uniform if the binary pattern contains at most two bitwise transitions from 0 to 1 or vice versa. For example, the patterns 00000000 (0 transitions), 01110000 (2 transitions) and 11001111 (2 transitions) are

uniform whereas the patterns 11001001 (4 transitions) and 01010010 (6 transitions) are not. The uniform ones occur more commonly in any image textures than the non uniform ones; therefore, the latter ones are neglected. The uniform ones yield only 59 different patterns. To create a rotation invariant LBP descriptor, a uniform pattern can be rotated clockwise P-1(P = no of bits in the pattern) times. Each rotation will give a distinct pattern and a decimal value. All these 8 patterns will be considered as a single pattern. Hence, instead of 59 bins, only 8 bins are needed to construct a histogram representing the local feature for a given local region. Once, the LBP local features for all regions of a face are extracted, they are concatenated into an enhanced feature vector. This method is proven to be a growing success. It has been adopted by many researchers, and has been successfully used for facial expression recognition (Ma and Khorasani, 2004: 1588-1595; Zhao and Pietikäinen, 2007: 915-928).

LPQ (Local Phase Quantization) is another LBP like descriptor. (Ojansivu and Heikkilä, 2008: 236-243) originally proposed the blur insensitive LPQ descriptor. The spatial blurring is expressed as multiplication of the original image with a point spread function (PSF) in the frequency domain. Phase is an invariant property of an image. The LPQ method is based upon the phase of the original image when the PSF is centrally symmetric. The LPQ method examines a local M×N neighborhood $N_x$ at each pixel position of image *f(x)* and extracts the phase information using the short-term Fourier transform de-fined by Equation (1.2), where $\omega_u$ is the basis vector of the 2-D Discrete Fourier transform at frequency u, and $f_x$ is another vector containing all $M^2$ image samples from $N_x$.

$$F(\mathbf{u},\mathbf{x}) = \sum_{y \in N_{\mathbf{x}}} f(\mathbf{x}-\mathbf{y}) e^{-j2\pi \mathbf{u}^T y} = w_{\mathbf{u}}^T f_{\mathbf{x}} \tag{1.2}$$

$$\mathbf{F}_{\mathbf{x}} = [F(\mathbf{u}_1,\mathbf{x}), F(\mathbf{u}_2,\mathbf{x}), F(\mathbf{u}_3,\mathbf{x}), F(\mathbf{u}_4,\mathbf{x})]\,. \tag{1.3}$$

$$q_j(\mathbf{x}) = \begin{cases} 1, & \text{if } g_j(\mathbf{x}) \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{1.4}$$

$$f_{\mathrm{LPQ}}(\mathbf{x}) = \sum_{j=1}^{8} q_j(\mathbf{x}) 2^{j-1}. \tag{1.5}$$

The local Fourier coefficients are at four frequency points: $u_1=[a,0]^T$ , $u_2=[0,a]^T$, $u_3=[a, a]^T$, and $u_4=[a,-a]^T$, where a is a sufficiently small scalar. The vector for each pixel is obtained using Equation (1.3). The phase information is acquired by Equation (1.4), which is a scalar quantizer. In the equation $g_j(x)$ is the *j*-th component of the vector $G_x=[Re\{F_x\},Im\{F_x\}]$. The resulting eight binary coefficients $q_j(x)$ are represented as an integer value between 0–255 using (1.5).

### 1.1.3 Facial Expression Classification

A classifier does this job. An algorithm that carries out classification, especially in a concrete implementation, is known as a classifier. The term 'classifier' sometimes also refers to some mathematical functions. The functions are carried out by a classification algorithm and maps input data to a class. Classification and clustering are examples of the more general problem of pattern recognition. Classification can be two types - binary classification and multiclass classification. In case of binary classification, only two classes are involved, whereas in multiclass classification there are more than two classes (Har-peled and Roth, 2003: 785-792). Since vast research has been done and classification procedures have been developed for binary classification, multiclass classification often needs to combine multiple binary classifiers to get a multiclass result. A wide range of classifiers have been applied to the automatic facial expression recognition problem: (Colmenarez, Frey and Huang, 1999: 592-597) implemented a Bayesian Recognition System (BRS) where they found the facial expression that maximizes the likelihood of a test image. (Matsuno et al., 1995: 352-359) classified expression by thresholding the Normalized Euclidean Distance (NED) in the feature space. Some other classification methods which had been used in facial expression recognition include Higher Order Singular Value Decomposition (Wang and Ahuja, 2003: 958-965), Fisher discrimination analysis (Shinohara and Otsuf, 2004: 499-504), Locally Linear Embedding (Wu and Lai, 2006) and so on (Aleksic and Katsaggelos, 2006: 3-11, Kotsia and Pitas, 2007: 172-187, Yin and Wei, 2006: 603-608). Among them the most successful ones are Support Vector Machine (Bartlett et al., 2003: 53; Michel and Kaliouby, 2003: 258-264; Xu et al., 2006: 309-312) etc. and Neural Net-work (Ichimura, Oeda and Yamashita, 2002: 2422-2427, Chang and Lin, 2011: 27; Kobayashi, Tange and Hara,

1995: 179-186; Ma and Khorasani, 2004: 1588-1595; Zhang et al., 1998: 454-459 etc.). All the above-mentioned methods are widely used in statistical learning. Only Support Vector Machine is introduced because it will be used in the proposed system.

1.1.3.1 Support Vector Machine

Support Vector Machine falls in the class of linear classifier, which maximizes the margin between two data classes. So it is also known as Optimal Margin Classifier (Boser, Guyon and Vapnik, 1992: 144-152). The Support Vector Machine (SVM) classifier is a powerful classifier that works considerably on a wide range of high dimensional data classification. The main disadvantage of the SVM algorithm is that it has several key parameters that need to be balanced to achieve the best results from a classification. For example, parameters that may give excellent classification accuracy for a given problem 'A' may give poor classification accuracy for problem 'B'. Therefore, the user may have to experiment with a number of different parameter settings in order to achieve a satisfactory result.

1) Linear SVM : separable case

Let the set of training examples $D$ be $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_r, y_n)\}$, where $\mathbf{x}_i = (x_1, x_2, \ldots, x_r)$ is an input vector in a real-valued space $X \subseteq R^r$ and $y_i$ is its class label (output value), $y_i \in \{1, -1\}$. 1: positive class and -1: negative class. SVM finds a linear function of the form (w: weight vector)

$$f(\mathbf{x}) = \langle \mathbf{w} \cdot \mathbf{x} \rangle + b \tag{1.6}$$

So that an input vector $\mathbf{x_i}$ is assigned to a positive class if $f(x_i) \geq 0$, and to the negative class if $f(x_i) < 0$.

$$y_i = \begin{cases} 1 & if \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 0 \\ -1 & if \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b < 0 \end{cases}$$

The hyperplane that separates positive and negative training data is,

$$\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0 \tag{1.7}$$

It is also called the decision boundary (surface). So many possible hyperplanes, SVM looks for the separating hyperplane with the largest margin. Machine learning theory says that this hyperplane minimizes the error bound.

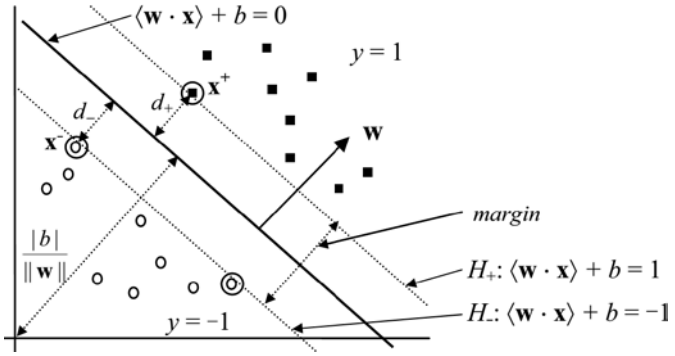



**Figure 1.6** Maximum-margin Hyper-planes for a SVM Trained with Samples from Two Classes, Support Vectors are Circled

**Source:** Liu, 2007: 111.

Assuming the data are linearly separable and considering a positive data point ($\mathbf{x}^+$, 1) and a negative ($\mathbf{x}^-$, -1) that are closest to the hyperplane $\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$, we may define two parallel hyperplanes, $H_+$ and $H_-$, that pass through $\mathbf{x}^+$ and $\mathbf{x}^-$ respectively and $H_+$ and $H_-$ are also parallel to $<\mathbf{w} \cdot \mathbf{x}> + b = 0$. The following equations can be obtained by rescaling $\mathbf{w}$ and $b$.

$$H_+: \langle \mathbf{w} \cdot \mathbf{x}^+ \rangle + b = +1 \tag{1.8}$$

$$H_-: \langle \mathbf{w} \cdot \mathbf{x}^- \rangle + b = -1 \tag{1.9}$$

Subject to: $\langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b \geq 1$ if $y_i = +1$

$\langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b \leq -1$ if $y_i = -1$

The distance between the two margin hyperplanes $H_+$ and $H_-$ is the margin ($d_+ + d_-$ in the Figure 1.6). Recall from vector space in algebra that the (perpendicular) distance from a point $\mathbf{x}_i$ to the hyperplane $\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$ is:

$$\frac{|\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b|}{\|\mathbf{w}\|} \tag{1.10}$$

Where ||**w**|| is the norm of **w**,

$$|| \mathbf{w} ||= \sqrt{< \mathbf{w} \cdot \mathbf{w} >} = \sqrt{w_1^{\,2} + w_2^{\,2} + ... + w_r^{\,2}} \tag{1.11}$$

To compute $d_+$ or $d_-$. , the distance from any point $\mathbf{x}_s$ on the plane $\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$ to the plane $\langle \mathbf{w} \cdot \mathbf{x}^+ \rangle + b = 1$ is computed by applying the Eq. (1.10) and noticing $\langle \mathbf{w} \cdot \mathbf{x}_s \rangle + b = 0$,

$$d_+ = d_- = \frac{|\langle \mathbf{w} \cdot \mathbf{x_s} \rangle + b - 1|}{|| \mathbf{w} ||} = \frac{1}{|| \mathbf{w} ||} \tag{1.12}$$

$$margin = d_+ + d_- = \frac{2}{|| \mathbf{w} ||} \tag{1.13}$$

Since SVM looks for the separating hyperplane that maximizes the margin which is the same as minimizing $||w||^2/2 = ‹w.w›/2$.

Given a set of linearly separable training examples, $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x_n}, y_n)\}$, learning is to solve the following constrained minimization problem,

$$\begin{aligned} &\text{Minimize: } \frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2} \\ &\text{Subject to: } y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1, \quad i = 1, 2, ..., n \end{aligned} \tag{1.14}$$

Note that $y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1, \quad i = 1, 2, ..., n$ summarizes

$\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq +1$ for $y_i = +1$

$\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \leq -1$ for $y_i = -1$.

This equation is a quadratic function subjects to linear constraints. Many algorithms exist for solving quadratic mathematical programming problems. The solution involves constructing a dual problem where a Lagrange multiplier $\alpha_i$ is associated with every constraint (or every training example) in the

primary problem:

$$Maximize: \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \mathrm{x_i} \cdot \mathrm{x_j} \rangle \tag{1.15}$$

$$Subject\ to: \quad \sum_{i=1}^{n} y_i\, \alpha_i = 0$$

$$\alpha_i \geq 0, \ \ i = 1,2, \ldots\ldots, n.$$

Where, $\alpha_i \geq 0$ are the Lagrange multipliers. The final decision boundary (maximal margin hyperplane) is:

$$\langle \mathrm{w} \cdot \mathbf{x} \rangle + \mathrm{b} = \sum_{\mathrm{i} \epsilon \mathrm{sv}} \alpha_i y_i \langle \mathrm{x_i} \cdot \mathbf{x} \rangle + b = 0 \tag{1.16}$$

Where, *sv* is the set of indecies of the support vectors which are the training examples with $\alpha_i > 0$. This equation is used for classification. To classify a test instance **z**, the decision function is:

$$f(\mathbf{z}) = \mathrm{sign}(\langle \mathrm{w} \cdot \mathbf{z} \rangle + b) = \mathrm{sign}\left(\sum_{\mathrm{i} \epsilon \mathrm{sv}} \alpha_\mathrm{i} y_\mathrm{i} \langle \mathrm{x_i} \cdot \mathbf{z} \rangle + b\right) \tag{1.17}$$

If it returns 1, then the test instance z is classifies as positive; otherwise, it is classified as negative.

2) Linear SVM : non-seprable case

In real world problems, linear separation of data from different classes may not be possible due to data distribution and noises. Noises cause large overlap of the data points from different classes. Therefore a linear SVM for non-separable case of data was suggested by (Aizerman et al., 1964: 821-837).

Recall in the separable case, the problem was,

$$\text{Minimize:} \quad \frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2}$$
$$\text{Subject to:} \ \ y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1, \quad i = 1, 2, ..., \mathrm{n} \tag{1.18}$$

With noisy data (errors), the constraints may not be satisfied so the margin constraints are relaxed by introducing slack variables, $\xi_i$ ($\geq 0$) as follows:

$$\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 1 - \xi_i \qquad \text{for } y_i = +1 \tag{1.19}$$

$$\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \leq -1 + \xi_i \qquad \text{for } y_i = -1 \tag{1.20}$$

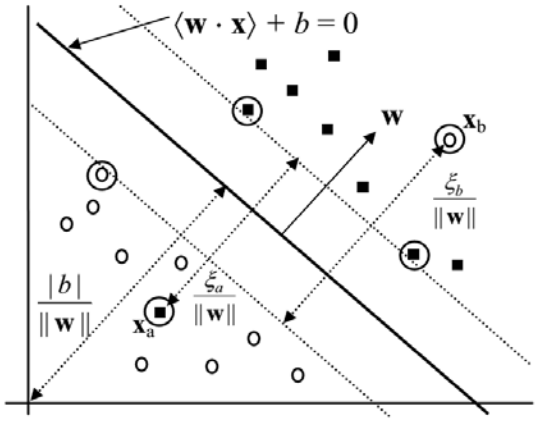



**Figure 1.7** The non-separable case: $x_a$ and $x_b$ are error data points.
**Source:** Liu, 2007: 118.

The new constraints:

$$\text{Subject to: } y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1 - \xi_i,\ i = 1, \ldots, n,$$
$$\xi_i \geq 0,\ \ i = 1, 2, \ldots, n.$$

The objective function is then changed by assigning an extra cost to penalize the errors as follows,

$$\text{Minimize: } \frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2} + C\left(\sum_{i=1}^{n} \xi_i\right)^k \tag{1.21}$$

$k = 1$ is commonly used, which has the advantage that neither $\xi_i$ nor its Lagrangian multipliers appear in the dual formulation. The new optimization problem becomes:

$$\begin{aligned} &\text{Minimize:} \quad \frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2} + C\sum_{i=1}^{n} \xi_i \\ &\text{Subject to:} \ y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1 - \xi_i, \quad i = 1, 2, ..., n \\ &\qquad\qquad \xi_i \geq 0, \quad i = 1, 2, ..., n \end{aligned} \tag{1.22}$$

This formulation is called the **soft-margin SVM**. The primal Lagrangian is

$$L_P = \frac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle + C\sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i [y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) - 1 + \xi_i] - \sum_{i=1}^{n} \mu_i \xi_i \tag{1.23}$$

Where, $\alpha_i$, $\mu_i \geq 0$ are the **Lagrange multipliers.** The dual problem can be rewritten as

$$\begin{aligned} &Maximize: \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \mathrm{x_i} \cdot \mathrm{x_j} \rangle \\ &Subject\ to: \quad \sum_{i=1}^{n} y_i \alpha_i = 0 \\ &0 \leq \alpha_i \leq C, \ \ i = 1,2, \ldots\ldots, n. \end{aligned} \tag{1.24}$$

This equation can be solved numerically and the resulting $\alpha_i$ values are then used to compute w and b. The final decision boundary is

$$\langle \mathrm{w} \cdot \mathbf{x} \rangle + \mathrm{b} = \sum_{\mathrm{i} \epsilon \mathrm{sv}} \alpha_\mathrm{i} y_\mathrm{i} \langle \mathrm{x_i} \cdot \mathbf{x} \rangle + \mathrm{b} = 0 \tag{1.25}$$

To classify a test instance **z**, the decision function, which is the same as the separable case, is used:

$$f(\mathbf{z}) = \mathrm{sign}(\langle \mathrm{w} \cdot \mathbf{z} \rangle + \mathrm{b}) = \mathrm{sign}\left(\sum_{\mathrm{i} \epsilon \mathrm{sv}} \alpha_\mathrm{i} y_\mathrm{i} \langle \mathrm{x_i} \cdot \mathbf{z} \rangle + \mathrm{b}\right) \tag{1.26}$$

The value of C is chosen by trying a range of values on the training set for optimum validation.

3) Non Linear SVM

The SVM formulations require linear separation. Real-life data sets may need nonlinear separation. To deal with nonlinear separation, the same formulation and techniques as for the linear case are still used. Only the input data is transformed into another space (usually of a much higher dimension) so that a linear decision boundary can separate positive and negative examples in the transformed space, the transformed space is called the **feature space**. The original data space is called the **input space.** The basic idea is to map the data in the input space $X$ to a feature space $F$ via a nonlinear mapping $\phi$,

$$\begin{aligned} &\phi : X \to F \\ &\mathbf{x} \mapsto \phi(\mathbf{x}) \end{aligned} \tag{1.27}$$

After the mapping, the original training data set $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$ becomes: $\{(\phi(\mathbf{x}_1), y_1), (\phi(\mathbf{x}_2), y_2), \ldots, (\phi(\mathbf{x}_n), y_n)\}$. Therefore, the decision function becomes:

$$f(\mathbf{z}) = \text{sign}(\langle \text{w} \cdot \emptyset(\mathbf{z}) \rangle + \text{b}) = \text{sign}\left(\sum_{\text{i}\epsilon\text{sv}} \alpha_\text{i} y_\text{i} \langle \emptyset(\text{x}_\text{i}) \cdot \emptyset(\mathbf{z}) \rangle + \text{b}\right) \tag{1.28}$$

A kernel function is a function that corresponds to a dot product of two feature vectors in some expanded feature space:

$$K(x_i, x_j) = \langle \emptyset(x_i) \cdot \emptyset(x_j) \rangle \tag{1.29}$$

Using this equation the decision function can be rewritten as:

$$f(\mathbf{z}) = \text{sign}(\langle \text{w} \cdot \emptyset(\mathbf{z}) \rangle + \text{b}) = \text{sign}\left(\sum_{\text{i}\epsilon\text{sv}} \alpha_\text{i} y_\text{i} K(x_i, \boldsymbol{z}) + \text{b}\right) \tag{1.30}$$

Some commonly used kernel functions:

(1) Linear Kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle$

(2) Polynomial Kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = \langle 1 + \mathbf{x}_i \cdot \mathbf{x}_j \rangle^p$

(p is the degree of the polynomial kernel)

(3) Radial basis function or RBF : $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\frac{\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2}{2\sigma^2})$

(σ is the distance between the two closest data points with different class labels)

(4) Sigmoid: $K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta_0 \mathbf{x}_i^T \mathbf{x}_j + \beta_1)$

($\beta o$ is a multiplicative parameter for the sigmoid kernel and $\beta_1$ is an additive parameter for the sigmoid kernel)

4) Multiclass SVM

A multiclass support vector machine can be built by combining multiple binary classifiers among the classes. When there are more than two classes, multiple binary classifier can be built between either

(1) One of the class and the rest (*one-against-all*) or

(2) Between every pair of classes (*one-against-one*)

Classification of new instances for the *one-against-all* case is done by a *winner-takes-all strategy*, in which the classifier with the highest output function assigns the class. The *one-against-one* approach follows a *max-wins voting strategy*, in which every classifier assigns the instance to one of the two classes, then the vote for the assigned class is increased by one vote, and finally the class with the most votes determines the instance classification. The one-against-one approach is chosen for this study simply because its training time is shorter.

## 1.2 Objectives

Geometric features are sensitive to the face shape and image resolution variations, whereas appearance-based features contain unnecessary information. Facial Action Coding System (FACS) involves more complexity due to facial feature detection and extraction procedures. Geometric shape based models face problem with on plane face transformation. Many researchers adopt LBP-local binary pattern

but it produces long histograms, which slow down the recognition speed. Although LBP features achieved high accuracy rates for facial expression recognition, LBP extraction can be time consuming.

The primary objective of the research of this dissertation is to design, implement and evaluate a novel facial expression recognition system. Particularly, a new, effective local feature representation for facial expression recognition is proposed. The system should have the following characteristics:

1) Automatic: The system should be fully automatic without manually labeling fiducial landmarks during both of the recognition process and the training process since semi-automatic is inconvenient and time-consuming.

2) Accuracy: The system should meet an overall recognition rate above 90% while maintaining a low false rate for each facial expression.

3) Robustness: The system should work successfully under different lighting conditions and cluttered backgrounds.

4) Runtime Performance: The system should be able to achieve better runtime performance than other previously proposed systems.

## 1.3 Thesis Overview

The remainder of this thesis is structured as follows:

**Chapter 2** reviews the state of the art of global facial expression recognition.

**Chapter 3** gives an overview of the proposed facial expression recognition system- architecture.

**Chapter 4** presents the details of the experiments conducted to evaluate the performance of the proposed method and discusses the experimental results.

**Chapter 5** summarizes the contributions and limitations of this thesis and introduces the focus of future research.

# CHAPTER 2

# LITERATURE REVIEWS

This chapter reviews some of the past works in processing and understanding facial expression. The basic modules of all FER systems are face detection and alignment, feature extraction, and classification. Almost all the past and recent works in FER are based on methods that implement these steps sequentially and independently.

## 2.1 Ahsan, Jabid and Chong (2013)

In this paper, a new appearance-based feature extraction technique LTP (Local Transitional Pattern) was used in conjunction with Gabor Filter for facial feature representation and Support Vector Machine for expression classification. They manually cropped the facial region from the whole image and divided it into 42(7x6) sub-images. First, they transformed the facial image using Gabor Filter and applied LTP on the transformed image. They calculated LTP for pixel ($x_c$,$y_c$) using

$$LTP_{P,R1,R2}(x_c, y_c) = \sum_{p=0}^{P-1} s(g_{p1} - g_c) \otimes s(g_{p2} - g_c) * 2^p \quad (2.1)$$

Where $g_c$ denotes the color intensity value of the center pixel ($x_c$,$y_c$), $g_{p1}$and $g_{p2}$ denotes the color intensity value of P equally spaced pixels on the circumference of a circle with radius $R_1$ and $R_2$ respectively. A detailed example of calculating LTP for the center pixel of a local 5x5 region is shown in Figure 2.1. They computed the histogram from each 42 sub-images and concatenated them to build the final feature vector of length 42,075. They evaluated their method on the Cohn-Kanade dataset using SVM for expression recognition and compared with traditional LBP and Gabor Filter based FER systems.

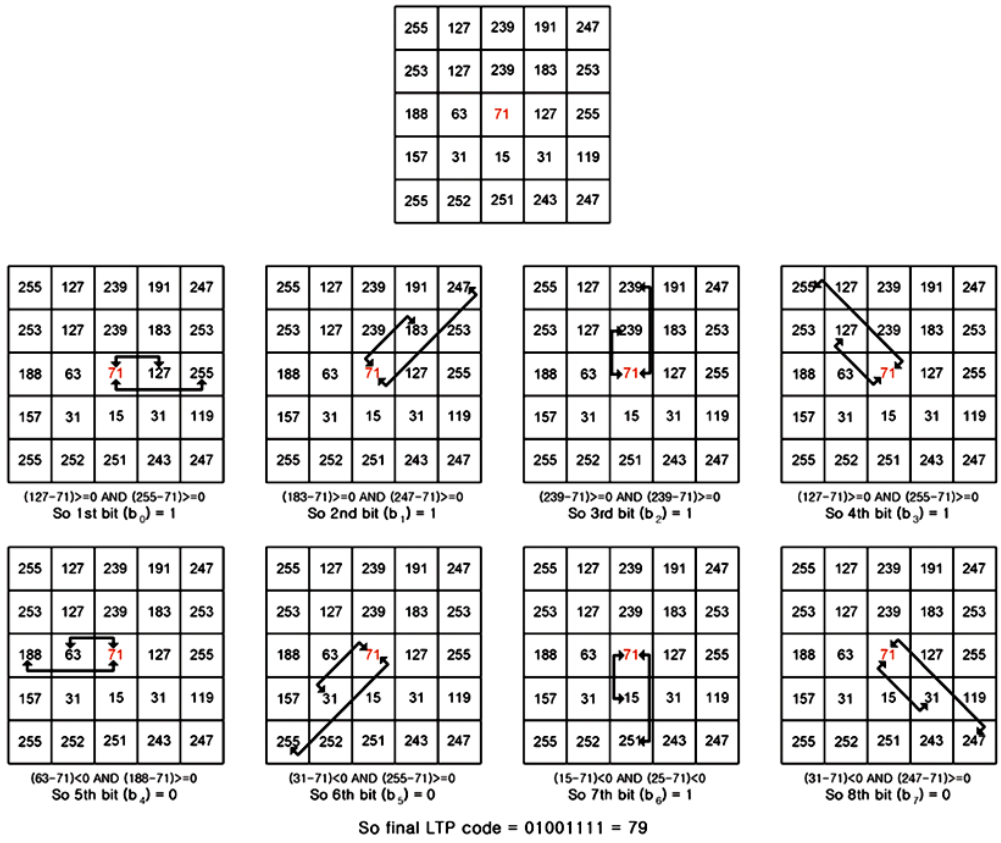



**Figure 2.1** Example of Obtaining LTP Pattern for the Center Pixel of a Local 5x5 Region with Radius 1 and 2

**Source:** Ahsan et al., 2013: 49.

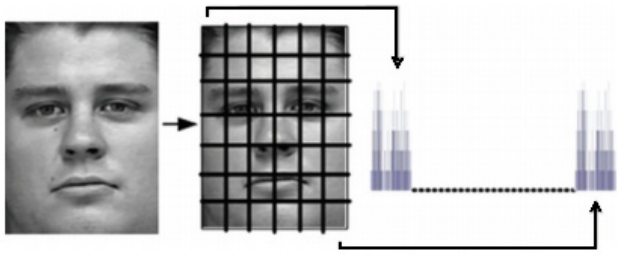


**Figure 2.2** Expression Image is Divided into Small Regions from Which Local Transitional Pattern Histograms are Extracted and Concatenated into Local Transitional Pattern Descriptor

**Source:** Ahsan et al., 2013: 50.

They claimed that though their system was little bit slow in performing facial expression recognition but the accuracy achieved was higher than those systems using

only LTP or LBP or Gabor Filter.

## 2.2 Kabir, Jabid and Chae (2012)

In this paper the authors used an appearance-based feature extraction technique LDPv (Local Direction Pattern Variance) and Support Vector Machine as an expression classifier.

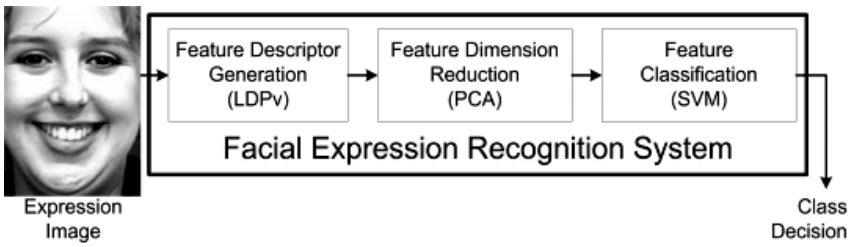



**Figure 2.3** Overview of Their Proposed System Based on LDPv
**Source:** Kabir et al., 2012: 383.

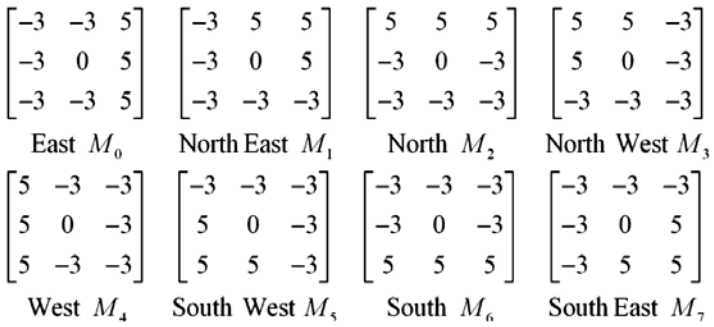



**Figure 2.4** Kirsch Edge Masks in all Eight Directions
**Source:** Kabir et al., 2012: 383.

To compute the LDPv code they first derived the LDP code from the 3x3 block. They multiplied the block with each of Kirsch edge masks (Figure 2.4) in all eight directions to get the edge response value for each surrounding pixel. They named the newly obtained 3x3 matrix as edge response matrix. In the edge response matrix, they replaced the top k-most value to 1 and rest to 0 and used those bits to build the LDP code. An example of obtaining LDP is shown in Figure 2.5 where k=3.

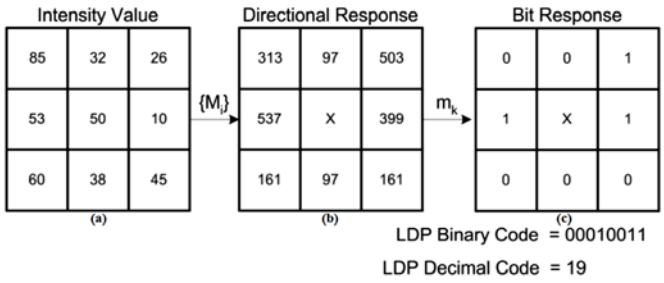



**Figure 2.5** Calculation of LDP Code with k=3
**Source:** Kabir et al., 2012: 384.

According to the authors, regions having high variance were more important than the low variance region. Therefore, they calculated variance for each 3x3 block in addition to the LDP pattern calculation and used the variance value as a weight for that block. In the time of building LDP histogram for the image, instead of adding 1 for a particular LDP pattern, they added the variance value for that code. They named this new technique of using variance with LDP code as LDPv.

They divided the whole image into 42 sub-images and concatenated LDPv histogram calculated from each sub-image. They evaluated their method on Cohn-kanade dataset with 6-class and 7-class facial expressions using both template matching and SVM as a classifier. Based on the experimental results, they claimed that LDP with local variance was more powerful than traditional LDP (Jabid, Kabir and Chae, 2010: 784-794) for facial expression recognition.

## 2.3 Yang and Bhanu (2012)

The overall system proposed in this paper is shown in Figure 2.6. The approach had four distinguished steps: 1) Face detection; 2) Face registration; 3) features extraction; and 4) the classification using a linear SVM-support vector machine classifier. They presented a new image-based representation called the emotion avatar image (EAI) and an associated reference image called avatar reference (Figure 2.7).

They condensed the image sequence from a video to form the EIA which was a single image.

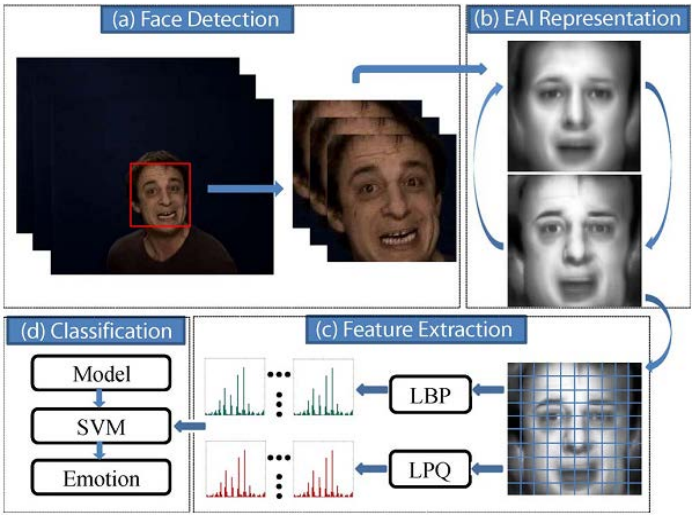



**Figure 2.6** Overall System Diagram of Yang and Bhanu's Approach
**Source:** Yang and Bhanu, 2012: 984.

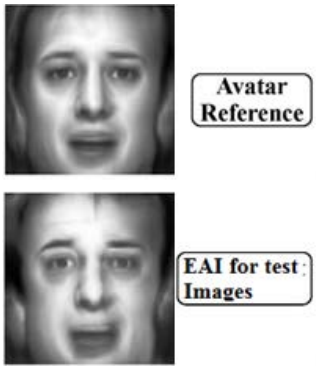



**Figure 2.7** Avatar referenced face model and EAI representations.
**Source:** Yang and Bhanu, 2012: 986.

EAI representation reduced the out-of-plane head rotation problem. It was robust to outliers and was able to gather dynamic information from expressions with different lengths. They used two appearance-based methods LBP and blur insensitive LPQ for feature extraction from the EAI image. Their system was tested on GEMEP-FERA dataset (video) and on extended Cohn-Kanade (CK+) dataset (static image). They proved that the information captured in an EAI (which was a single image rather than the whole image sequence from the video) was very effective for facial expression recognition.

## 2.4 Huang et al. (2011)

Huang Xiaohua *et al.* proposed a weighted-component based feature descriptor for expression recognition from video sequences.

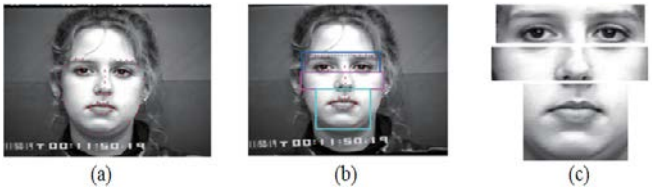



**Figure 2.8** (a) 62 Facial Points (dots) Derived by AAM (b) Rectangles Around the Mouth, Nose and Eyes Determined by 62 Facial Points (c) Cropped Eyes, Nose and Mouth

**Source:** Huang et al., 2011: 3.

They extracted both geometric and appearance-based features from three facial regions (Figure 2.8). They used AAM-Active Appearance Model to detect 62 fiducial points on face areas and cropped those areas e.g. mouth, nose and eyes, separately using those points. Then they used appearance-based method LBP-TOP (Local Binary Pattern on Three Orthogonal Planes) to extract the features from three cropped facial regions. They also computed dynamic features from those three regions using edge map (Figure 2.9)

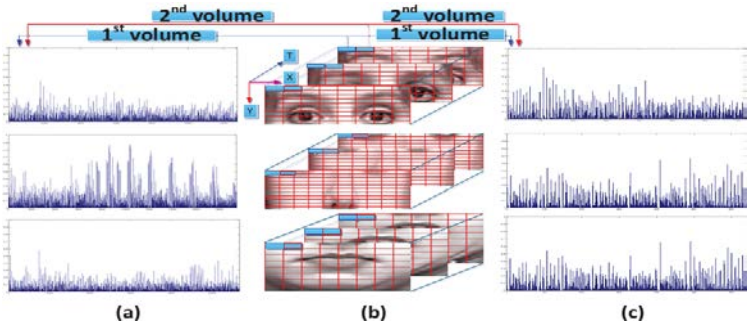



**Figure 2.9** Framework of Feature Extraction (a) Dynamic Appearance Representation by LBP-TOP (Local Binary Pattern on Three Orthogonal Planes); (b) Three Components (Eyes, Nose, Mouth); (c) Dynamic Shape Representation by Edge Map

**Source:** Huang et al., 2011: 4.

Motivated by an aforementioned MKL (Multiple Kernel Learning) method, they formulated a new automatic weight-learning method that can learn weights for multiple feature sets in facial components (Figure 2.10).

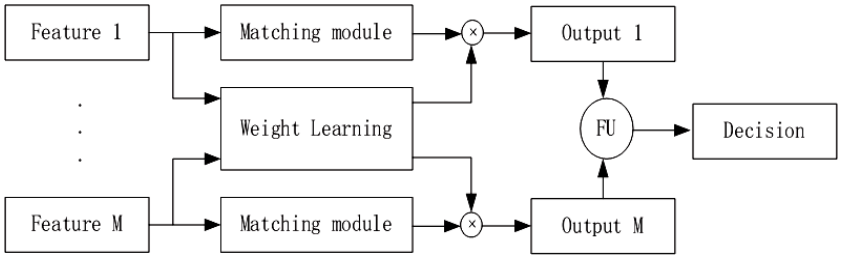


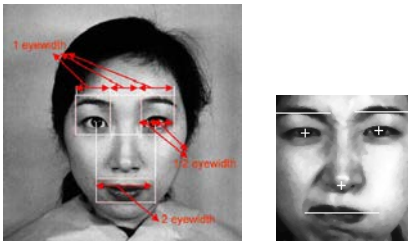

**Figure 2.10** Framework of Multiple Feature Fusion, FU: Fusion Module

**Source:** Huang et al., 2011: 5.

They conducted experiments on both person dependent and person independent environments. They evaluated their new strategy of automatic-weighted hybrid feature with feature fusion on the Extended Cohn-Kanade dataset and found that their method was more efficient than other state of art methods for facial expression recognition.

## 2.5 Liu, Li and Wang (2009, 2011)

The authors proposed an algorithm of automatic facial expression recognition for static images based on appearance-based method Local Binary Patterns on local



(a) (b)

**Figure 2.11** (a) The Facial Components Relations, (b) The Positions of Eyeballs Using Projection Method

**Source:** Liu et al., 2009: 198, 2011: 415.

areas for feature extraction and template matching for facial expression recognition.

In the preprocessing phase of expression recognition, they normalized all the gray scale images to 108 by 133 pixels based on the eyeballs, which were detected by projection method. Using this eyeball location information, they calculated manually the location of mouth through the pre-knowledge of face structure (Figure 2.11). They applied Local Binary Patter on each local areas e.g. eyes and mouth, and named it as LLBP technique. They divided the the local regions into non-overlapping rectangular sub-regions and computed histograms for each of the regions which is considered as the final feature for expression recognition. They followed template matching for recognition where they choose two images of each expression from each subject for training and rest of the images for testing. Their proposed system framework is shown in Figure 2.12.

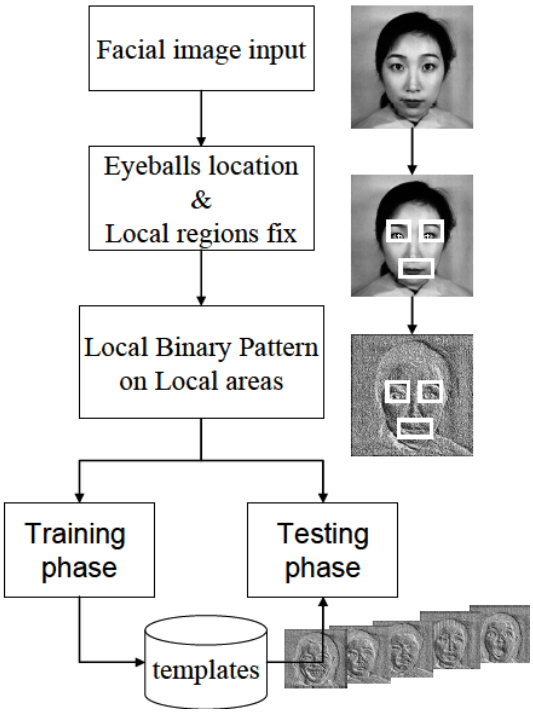



**Figure 2.12** Framework of Automatic Facial Expression Recognition System
**Source:** Liu et al., 2009: 199.

The experimented their methodology on the JAFFE dataset and compared classification accuracy of each expression class with the traditional LBP (Table 2.1). They found the accuracies were more prominent than the traditional LBP.

**Table 2.1** Recognition Rate Comparison of LLBP with Traditional LBP Class by Class

| Expressions | Traditional LBP (%) | LLBP (%) |
|---|---|---|
| Anger | 75 | 75 |
| Disgust | 57.5 | 80 |
| Fear | 62 | 65 |
| Happy | 74 | 83 |
| Sad | 52.5 | 66 |
| Surprise | 77.5 | 92 |

## 2.6 Huang et al. (2010)

In this paper, the authors proposed a component-based feature descriptor approach for facial expression recognition from video sequences. They extracted 38 important facial interest points, based on prior learning and information inspired by the methods presented in (Heisele and Koshizen, 2004:153-158) and (Lowe, 2004: 91-110) (Figure 2.13).

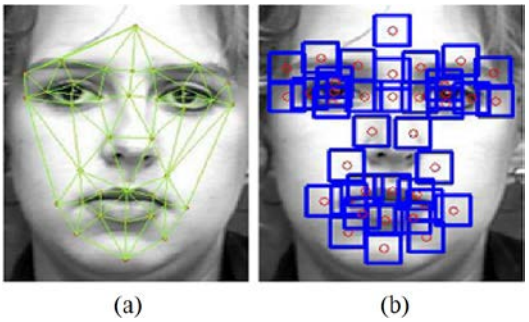



**Figure 2.13** (a) 38 Important Facial Interest Points (b) Regions around 38 Important Facial Interest Points for Feature Extraction

**Source:** Huang et al., 2011.

They created a square region of size 32x32 pixels around those 38 points for feature extraction, where the important points are in the center of each square. Most of the square regions are near to the eyes and mouth area (Figure 2.13(b)). They extracted features from each square using LBP-TOP (local binary pattern from three orthogonal planes) which was proved to be effective for appearance, vertical and horizontal motion in image sequence (Figure 2.14).

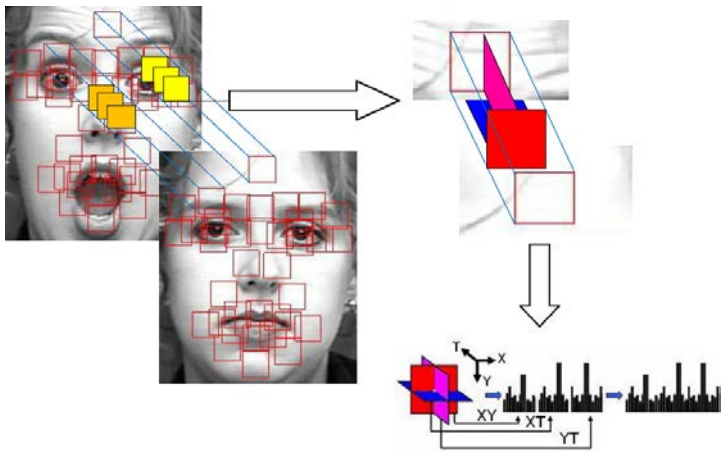
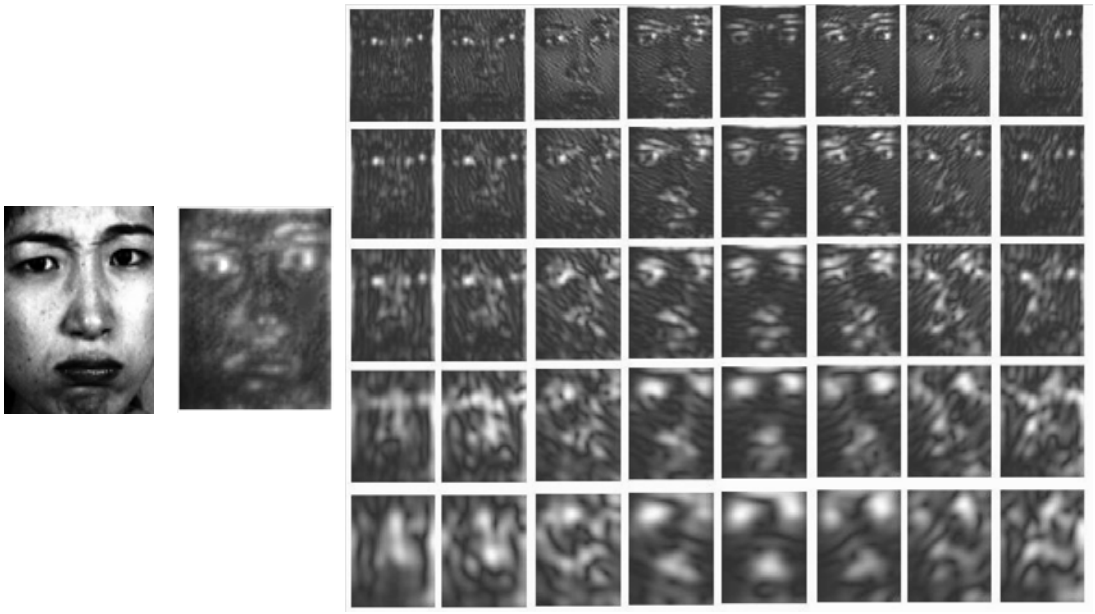



**Figure 2.14** Component Based Spatiotemporal Features in three Orthogonal Planes
**Source:** Huang et al., 2011.

They also incorporated AdaBoost (Adaptive Boosting) to reduce the feature dimension by selecting the most discriminative features for all the components. They used multi-classifier fusion, a new framework for fusing recognition results from several classifiers, such as support vector machines, boosting, fisher discriminant classifier, for expression recognition. Extensive experiments on the Cohn-Kanade facial expression database (Kanade, Cohn and Tian 2000: 46-53) were carried out to evaluate the performance of the proposed approach and they concluded that their approach of component-based spatiotemporal feature (CSF) extraction with multiple classifiers fusion as a classifier was more accurate than other state of art FER systems.

## 2.7 Lajevardi and Lech (2008)

Lajevardi et al. (2008: 1-6) proposed an automatic facial expression recognition method. A knowledge-based method was used to crop the facial region from the images. The formula used to do that was 2.2d × 1.8d where d is the distance between the two eyes and 2.2d × 1.8d is the size of the cropped rectangle. A set of characteristic features obtained by averaging the outputs from a Gabor Filter Bank with 5 frequencies and 8 different orientations was used to build the feature vector, see Figure 2.15(c). Gabor Filter is a holistic appearance-based feature extraction method but the main problem is its feature vector dimensionality.



**(a)** **(b)** **(c)**

**Figure 2.15** (a) Original Image, (b) An Averaged Gabor Filter Bank, (c) Gabor Filter Bank Feature Images in 8 Different Orientations

**Source:** Lajevardi et al., 2008: 74.

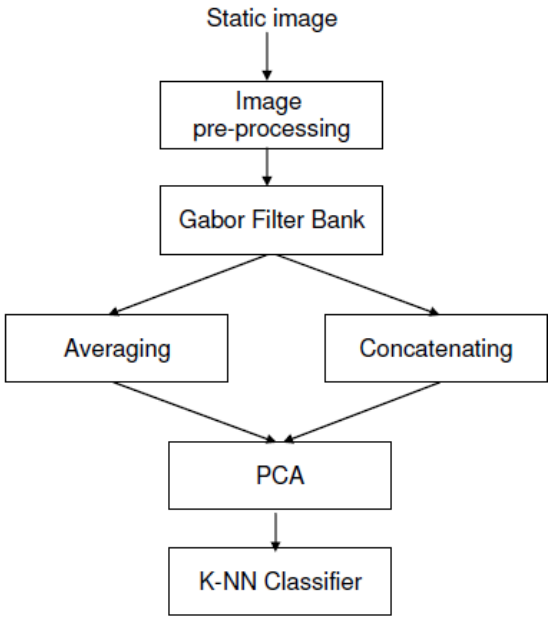



**Figure 2.16** Block Diagram of Facial Expression Recognition System Using Both the Full and Average of the Gabor Filter

**Source:** Lajevardi et al., 2008: 72.

In this paper the authors adopted and Principal Component Analysis (PCA) to reduce the feature dimensionality. The expression recognition tasks were performed using the K-Nearest Neighbor (K-NN) classifier. Extensive experimental results on publicly available dataset JAFFE showed that the Average Gabor Filter (AGF) achieved very high computational efficiency at the cost of a relatively small decrease in classification accuracy when compared to the full Gabor Filter features. Block diagram of facial expression recognition system using both the full and average Gabor Filter is shown in Figure 2.16.

## 2.8 Sun et al. (2008)

Sun *et al.* (2008) proposed a FER system based on appearance-based Local Gabor Binary Patterns (LGBP) for feature extraction and Support Vector Machine as a classifier on static images. The Gabor Coefficients Maps (GCMs) were extracted by convolving the face image (Figure 2.17) with Gabor Filters (Figure 2.18).

**Figure 2.17** Original Facial Image
**Source:** Sun et al., 2008: 159.

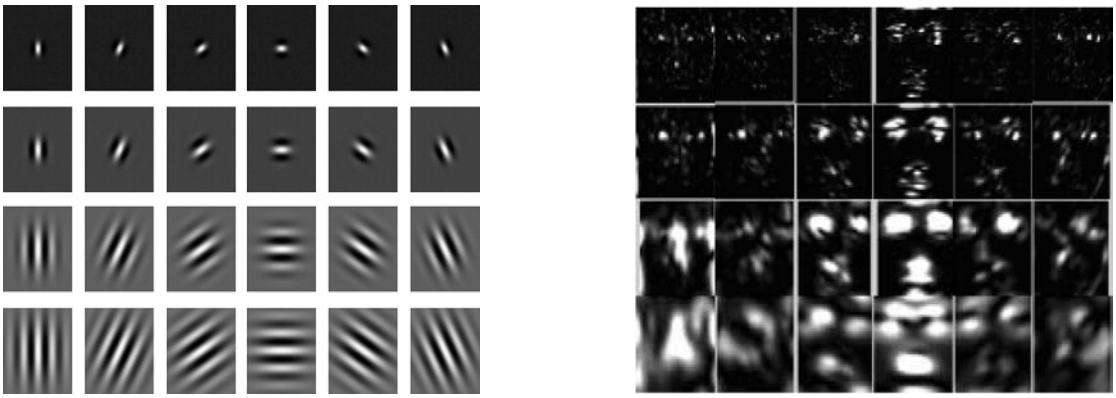


**Figure 2.18** Gabor Filter Set (Left), Gabor Features of the Face (Right)
**Source:** Sun et al., 2008: 160.

$$G(x, y, v, u) = G_k(x, y) * I(x, y), \quad (2.2)$$

Where, $G_k(x, y)$ is the Gabor filter and $I(x, y)$ is the face image. They performed LBP on GCM instead of the original image to reduce the feature vector dimension. The technique was named as LGBP.

$$LGBP=\sum_{p=0}^{8} G_p(x, y, v, u) - G_c(x, y, v, u), \quad (2.3)$$

Where p= 0 to 8 and c is the center of local 3x3 pixels region.

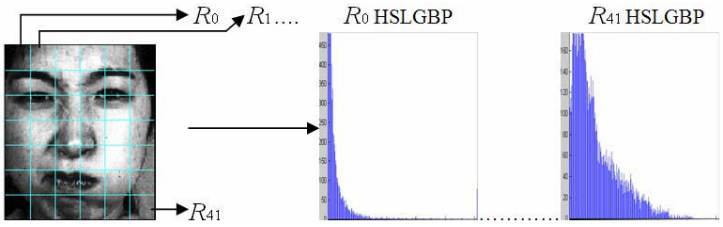



**Figure 2.19** The Computation of HSLGBP (Histogram)
**Source:** Sun et al., 2008: 161.

They divided the facial image into 42 sub-images and concatenated the histograms of LGBP codes calculated from each sub-images (

Figure **2.19**). Finally, the multi-class Support Vector Machine (SVM) was used to perform the feature classification. They claimed that their method LGBP with SVM was more efficient accurate than tradition LBP plus SVM.

## 2.9 Kotsia and Pitas (2007)

The facial expression recognition system developed in this paper used geometric features and it was semi-automated in the sense that the authors used some Candide grid nodes and manually placed them onto face landmarks to create a facial-wire-frame model for each facial expression (Figure 2.20 and Figure 2.21).

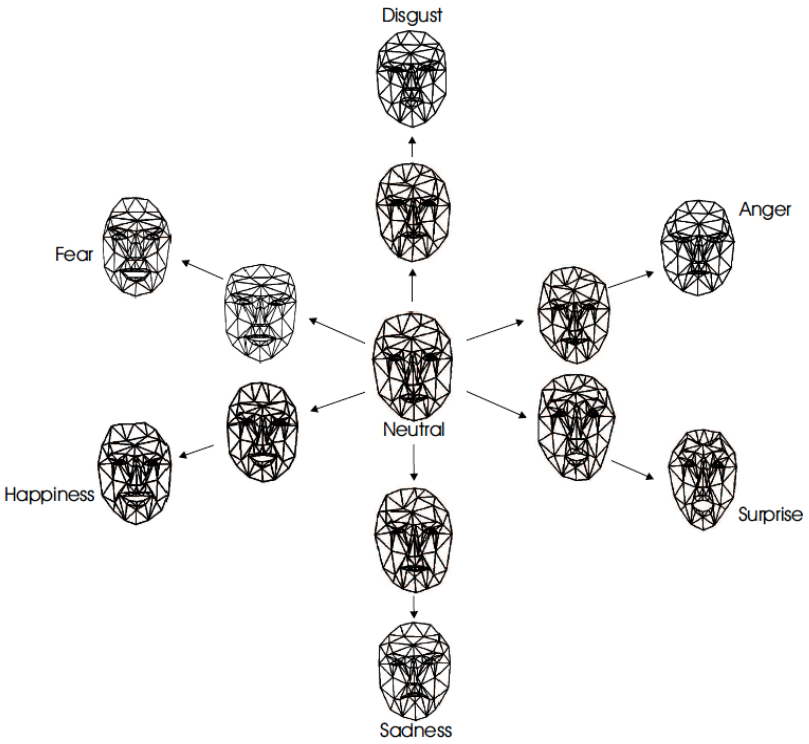



**Figure 2.20** An Example of the Deformed Candide Grids for Each One of the 6 Facial Expressions

**Source:** Kotsia and Pitas, 2007: 176.

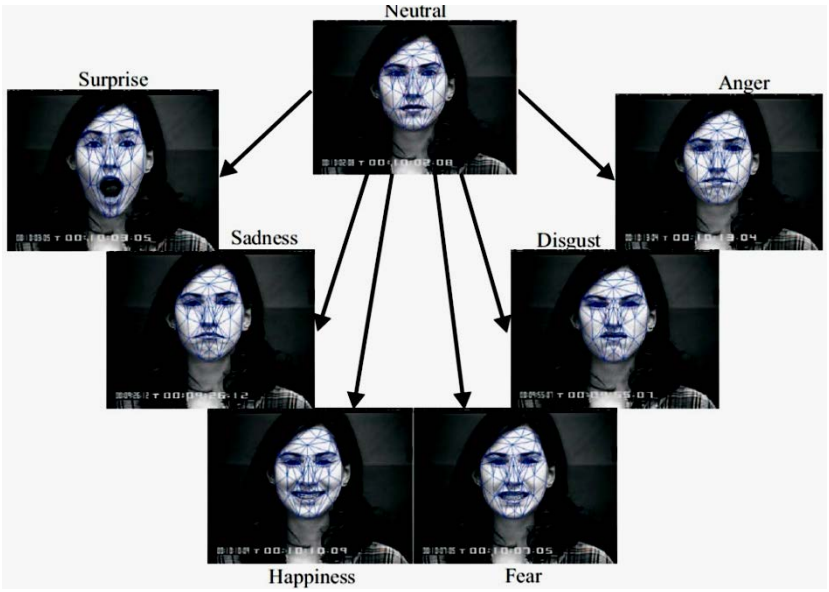



**Figure 2.21** Peak Expression with Candide Grid of a Single Subject

**Source:** Kotsia and Pitas, 2007: 176.

The Candide grid nodes were placed in the first frame onto the face and grid deformation was measured by deducting the first frame grid positions from the last frame grid positions (Figure 2.22). This geometrical displacement, defined as their coordinate difference was used as an input to the Support Vector Machine.

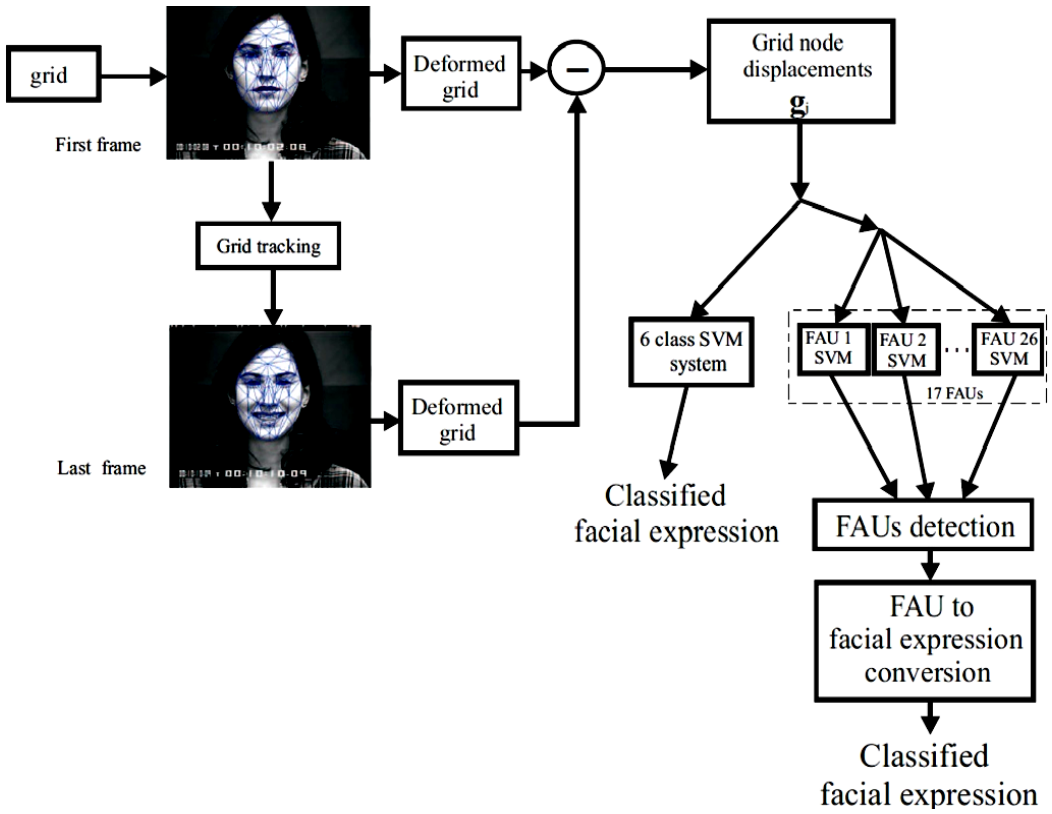



**Figure 2.22** System Architecture for Facial Expression Recognition in Facial Videos
**Source:** Kotsia and Pitas, 2007: 176.

The full system architecture for facial expression recognition is shown in figure Figure 2.22. The authors in this paper proposed two different methods for expression recognition, either by using the SVM directly or by detecting Facial Action Units (FAUs) then using SVM. In the first method of facial expression recognition, the SVM was composed of six six-class SVMs, one for each one of the 6 basic facial expressions recognition. In the second method, the SVMs system were consists of 8 two-class SVMs, one for each one of the 8 chosen FAUs was used. They obtained very good classification for both the above methods on CK dataset.

# CHAPTER 3

# PROPOSED SYSTEM ARCHITECTURE

## 3.1 Proposed System Framework

The overall system architecture developed in this work is shown in Figure 3.1.

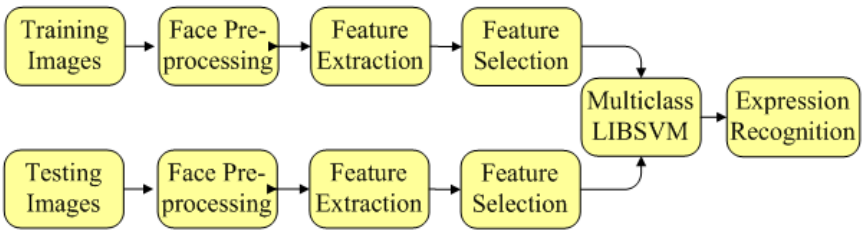



**Figure 3.1** Overall System Architecture

(1) For each of training images, convert it to gray scale if in different format.

(2) Detect the face in the image, resize it and divide it into equal size blocks.

(3) Compute feature value for each pixel using feature extraction method.

(4) Construct the histogram for each block.

(5) Concatenate the histograms to get the feature vector for each image.

(6) Build a multiclass Support Vector Machine for face expression recognition using feature vectors of the training images.

(7) Do step 1 to 5 for each of testing images and use the Multiclass Support Vector Machine from step 6 to identify the face expression of the given testing image.

Almost all the steps are based on statistical methods and SVM will be trained

using a labeled training set. The proposed system has three major phases, (a) Image Preprocessing, (b) Feature Extraction and (c) Classification. Under this framework, the performance of various existing algorithms will be compared with proposed methods to find out the optimal configuration of a FER system. Phase (b) is the major contribution of this work.

## 3.2 Image Preprocessing

This module consists of three components:

1) Face detection,
2) Face masking, and
3) Face normalization.

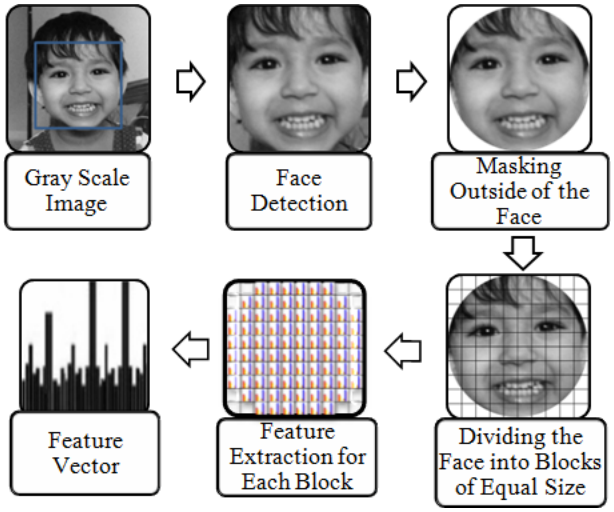
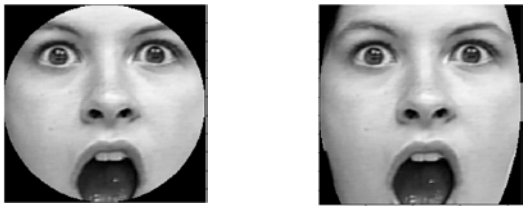



**Figure 3.2** Steps of Facial Feature Extraction

Face detection is done using '*fdlibmex*' library from the Matlab software. It is a very simple face detection library for matlab. No toolboxes are required. The library consists of a single '*mex*' file with a single function that takes an image as input and outputs the locations of the frontal faces in the image of varying dimension. Unfortunately, there are no further explanations or a reference from the author as this is a piece of unpublished work. The method is found quite an effective tool and it is worth to investigate into its performance. In face masking, unwanted facial areas,

which do not have any role for facial expression e.g. hair, both neck sides are removed using Different shapes like round or elliptical as shown in Figure 3.3.



(a) (b)

**Figure 3.3** Sample Face (a) Masked Using Round Shape, (b) Masked Using Elliptical Shape

The outside of the shape is multiplied using ‘NaN’ stands for not a number, is a numeric data type value representing an undefined or unrepresentable value. The elliptical shape is more preferable as for global case it covers more informative area of the face than round one. In face normalization, face shape is re-dimensioned to a fixed size near to original face dimension for better performance.

## 3.3 Feature Extraction

To ease the task of the classifier and achieve better classification accuracy, facial images are transformed to feature vector by projecting them into a feature space, which is called feature extraction. Feature extraction simplifies the amount of necessary data required to describe a large set of facial data accurately. When performing analysis of complex data, a major problem is the number of variables involved with that data. Analysis data with a large number of variables usually requires a large amount of memory and computational power. Feature extraction is a general term for methods of creating combinations of the variables to get around these problems while still describing the data with adequate information. Variables are sometimes denoted as bin in Facial Expression Recognition Systems (FERS). Each variable or bin represents particular type of facial characteristics. This module of the proposed system is mainly focused on the feature extraction and selection task, so when designing the classifier one does not need to think about the classifier input. A

novel feature-extraction method Gradient Direction Patter is proposed and tried in different ways in this work.

They are named as:

(1) GDP-2a,

(2) GDP-2b,

(3) GDP-4 and

(4) GDP-12

### 3.3.1 GDP-2a

This method uses only the color values of a pixel and its four neighboring pixels, i.e. North, West, South and East directions, to compute the local pattern for the pixel, see Figure 3.4.

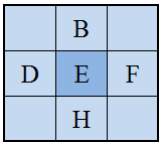



**Figure 3.4** Considered Pixels for GDP-2a

The pattern can be derived as follows:

$$gd(1) = (f - d) \tag{3.1}$$

$$gd(2) = (b - h) \tag{3.2}$$

$$D(i) = \begin{cases} 0 & \text{if} \quad gd(i) < 0 \\ 1 & \text{if} \quad gd(i) \geq 0 \end{cases} \tag{3.3}$$

Where b, d, e, f and h are the gray color intensity values of neighboring pixels of the current pixel E i.e. B, D, E, F and H, respectively. The gd(i) represents the gradient value between gray color intensities of two opposite neighboring pixels of the current pixel for the i-th direction. The D(i) corresponds to the gradient direction for the i-th direction. Thus, the binary vector of D contains 2 bits representing $2^2$=4 different patterns. Therefore, the GDP feature vector length for each block is 4. A detailed example of the gradient direction pattern extraction at pixel E is given in

Figure 3.5.

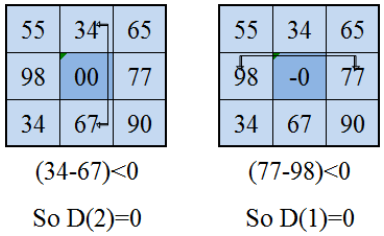



**Figure 3.5** Example for Computing GDP-2a

Due to its tiny feature vector length, this method is good for large-scale facial dataset. Though it counts only four pixels among the eight neighboring pixels, in terms of accuracy it performs noticeably well.

### 3.3.2 GDP-2b

This method is similar to GDP-2a except that the four neighboring pixels in North-West, North-East, South-West and South-East directions are considered here, see Figure 3.6.

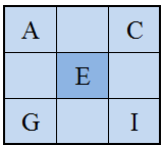



**Figure 3.6** Considered Pixels for GDP-2b

A detailed example of the gradient direction pattern extraction at pixel E is given in Figure 3.7.

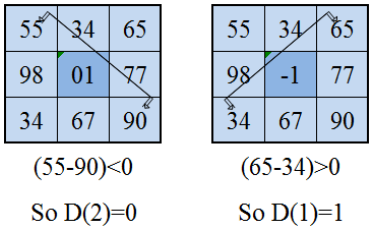



**Figure 3.7** Example for Computing GDP-2b

Therefore, GDP-2a captures color gradient over horizontal and vertical directions; on the other hand, GDP-2b captures two corner directions.

### 3.3.3 GDP-4

GDP-4 considers all four possible gradient directions through the center pixel in a 3x3 pixels region e.g. AI, BH, CG and FD (Figure 3.8).

| | | |
|---|---|---|
| A | B | C |
| D | E | F |
| G | H | I |

**Figure 3.8** Considered Pixels for GDP-4

Gradient directions form GDP-2a and GDP-2b are combined here to formulate a new 4-bit binary pattern. The pattern can be derived as follows:

$$gd(1) = (f - d) \tag{3.4}$$

$$gd(2) = (c - g) \tag{3.5}$$

$$gd(3) = (b - h) \tag{3.6}$$

$$gd(4) = (a - i) \tag{3.7}$$

$$D(i) = \begin{cases} 0 & \text{if} \quad gd(i) < 0 \\ 1 & \text{if} \quad gd(i) \geq 0 \end{cases} \tag{3.8}$$

Where a, b, c, d, f, g, h and i are the gray color intensities of neighboring pixels of the current pixel E, i.e. A, B, C, D, F, G, H and I, respectively. The gd(i) represents the gradient value between gray color intensities of two opposite neighboring pixels of the current pixel for the i-th direction. The D(i) corresponds to the gradient direction for the i-th direction.

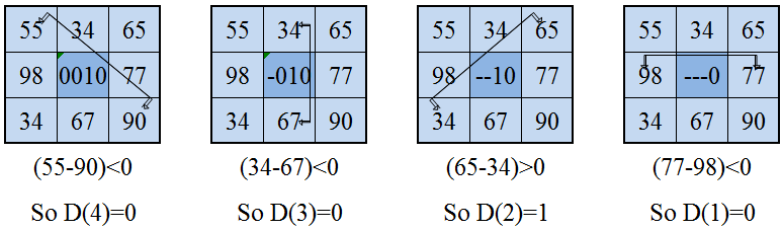



**Figure 3.9** Example for Computing GDP-4

Thus, the binary vector of D contains 4 bits representing $2^4$=16 different patterns. Therefore, the GDP feature vector length for each block is 16. A detailed example of the gradient direction pattern extraction at pixel E is given in Figure 3.9. Therefore, GDP-4 captures color gradient over horizontal and vertical and two corner directions.

### 3.3.4 GDP-12

GDP-12 considers larger area than GDP-4 to compute local feature for a pixel (Figure 3.10).

| | | | | |
|---|---|---|---|---|
| A2 | B2 | C2 | D2 | E2 |
| F2 | A1 | B1 | C1 | G2 |
| H2 | D1 | E | F1 | I2 |
| J2 | G1 | H1 | I1 | K2 |
| L2 | M2 | N2 | O2 | P2 |

**Figure 3.10** Considered Pixels for GDP-12

Two separate patterns are calculated from 5x5 pixels region. First pattern is the GDP-4, which is calculated from the shaded area of the Figure 3.10. The second pattern is an 8-bit pattern, which is derived from the non-shaded area of Figure 3.11 as follows:

$$gd(1) = (k2 - f2) \tag{3.9}$$

$$gd(2) = (i2 - h2) \tag{3.10}$$

$$gd(3) = (g2 - j2) \tag{3.11}$$

$$gd(4) = (e2 - l2) \tag{3.12}$$

$$gd(5) = (d2 - m2) \tag{3.13}$$

$$gd(6) = (c2 - n2) \tag{3.14}$$

$$gd(7) = (b2 - o2) \tag{3.15}$$

$$gd(8) = (a2 - p2) \tag{3.16}$$

$$D(i) = \begin{cases} 0 & \text{if} \quad gd(i) < 0 \\ 1 & \text{if} \quad gd(i) \geq 0 \end{cases} \tag{3.17}$$

Where a2, b2, c2, d2, e2, f2, g2, h2, i2, j2, k2, l2, m2, n2, o2 and p2 are the gray color intensities of neighboring pixels of the current pixel E, i.e. A2, B2, C2, D2, E2, F2, G2, H2, I2, J2, K2, L2, M2, N2, O2 and P2 respectively. The gd(i) represents the gradient value between gray color intensities of two opposite second level neighboring pixels of the current pixel for the i-th direction. The D(i) corresponds to the gradient direction for the i-th direction. Thus, the binary vector of D for the second level contains 8 bits representing $2^8$=256 different patterns. Therefore, the GDP feature vector length for each block is 16+256=272. A detailed example of the gradient direction pattern extraction at pixel E is given in Figure 3.11.

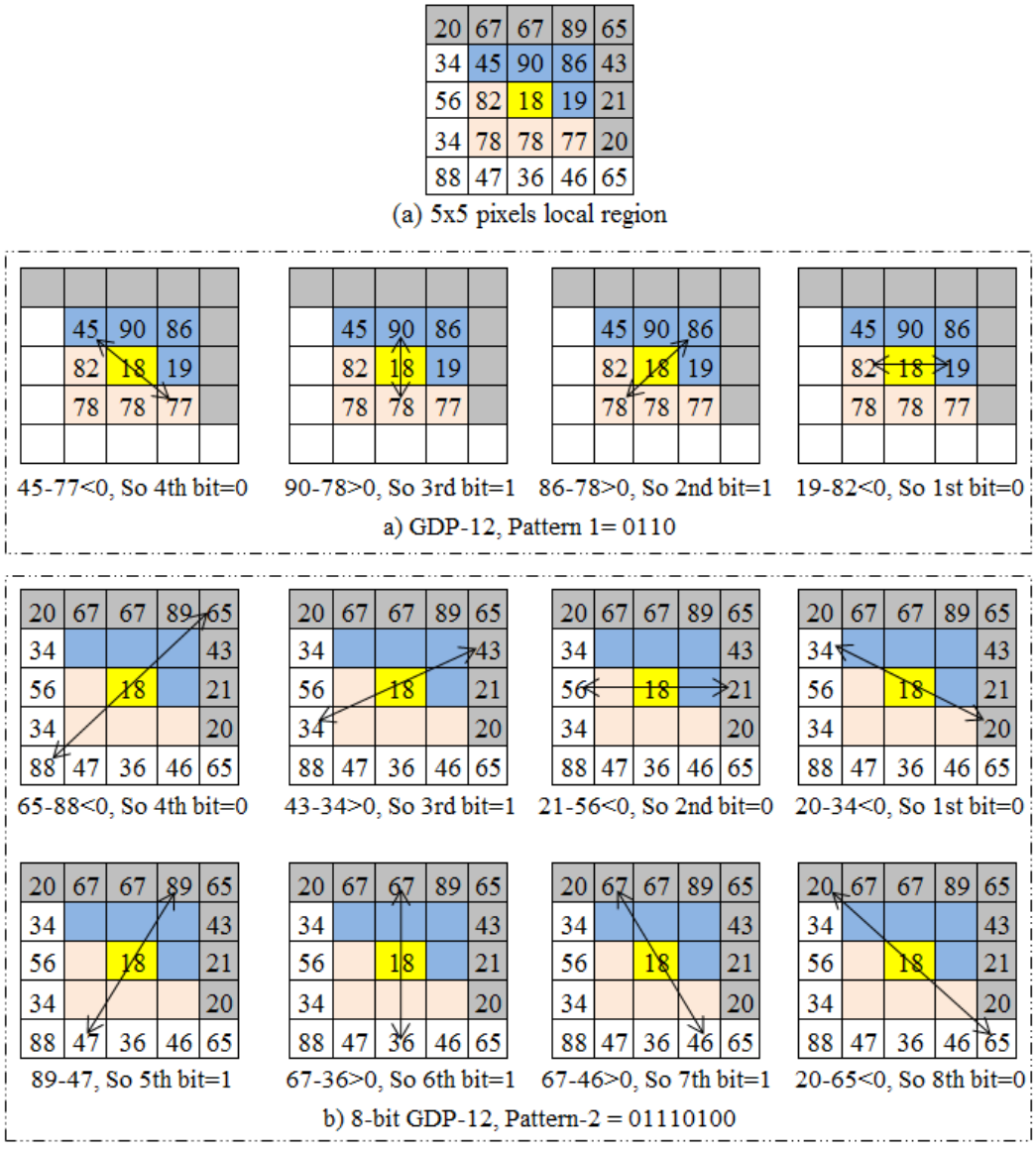



**Figure 3.11** Example for Computing GDP-12

A histogram H contains occurrence counts for all possible GDP patterns at all pixels of the input image I of size *h by w*. Each occurrence count of a GDP pattern P, count(P), can be calculated using equation (3.18). The resultant histogram is the GDP descriptor for that image.

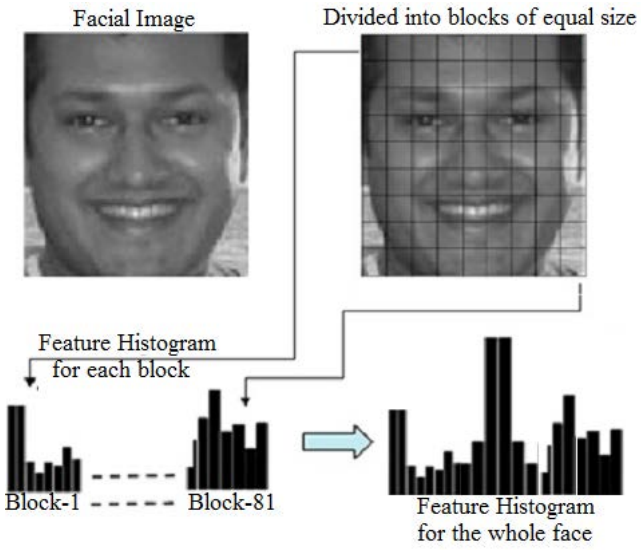



**Figure 3.12** Facial Feature Extraction

$$\text{Count}(P) = \sum_{x=1}^{h} \sum_{y=1}^{w} f(GDP(x,y), P) \tag{3.18}$$

$$Where, f(GDP(x,y), P) = \begin{cases} 1 & GDP(x,y) = P \\ 0 & else \end{cases}$$

The GDP histogram computed from the whole image misses spatial information on locations where GDP patterns occur. However, for facial expression recognition, this information is quite important for distinquishing the facial expressions. Hence, the basic histogram is modified to an extended histogram, where the input image I is divided into *N* number of blocks e.g. $B_1$,$B_2$,…, $B_N$, and the *GDP* histogram $H_i$ is built for each block $B_i$ , where *i= 1,2,....., N* (Number of blocks).

Finally, concatenating all the $H_i$ yields the feature vector of size *NxV* where, *V* is the length of each histrogram (the number of possible GDP patterns). This extended feature histogram represents local feature with some degree of the spatial information.

## 3.4 Gray-Scale Invariant Property of the GDP

The GDP feature representation can be shown to have the gray-scale invariant property. Let suppose a grayscale image is linearly transformed using the following equation,

$$g^t(x,y) = \alpha\, g(x,y)+\beta;$$

Where, g(x,y) is the original gray color value of the pixel with the coordinate of (x,y), α is a scale factor, β is a shift factor and $g^t(x,y)$ is the gray color value of the pixel after transformation. The transformation may be due to the change on illumination condition.

If the two GDP patterns derived for any particular corresponding pixel of the original image and the transformed one are the same then it can be concluded that the GDP feature representation is indeed grayscale invariant. Due to proximity, it can be assumed that all pixels within a local block, either 3x3 pixels for GDP-2a, GDP-2b and GDP-4 or 5x5 pixels for GDP-12, have approximately the same values of the scale factor and the shift factor.

The GDP pattern at the center pixel of a block of the original image can be derived as a binary vector as follows,

$$GDP = [\, S(A_0-B_0),\ S(A_1-B_1), \ldots, S(A_{n-1} - B_{n-1}) \,]$$

Where, $A_i$ and $B_i$ represent the gray color values of the two neighboring pixels considered for the calculation of the gradient direction i, n is the number of considered directions for the respective GDP, and S is a sign function defined as follows

$$S(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

After varying illumination, the new transformed GDP at the center pixel can be derived as follows,

$$GDP^t = [S((\alpha A_0+\beta) - (\alpha B_0+ \beta)), S((\alpha A_1+\beta) - (\alpha B_1+ \beta)),.,S((\alpha A_{n-1}+\beta) - (\alpha B_{n-1}+ \beta))]$$

$GDP^t = [S(\alpha ( A_0 - B_0 )), S(\alpha (A_1 - B_1 )),..,S(\alpha ( A_{n-1} - B_{n-1}))]$

The shift factor β is eliminated from the calculation of the new $GDP^t$ and the scaling factor does not change the sign values or the gradient directions for all the components from the original GDP since $S(\alpha ( A_i – B_i )) = S(A_i – B_i)$ for all i.

Hence, $GDP^t = GDP$

Therefore, the GDP feature representation is proven to be gray scale invariant and is not subject to the illumination variations.

## 3.5 Feature Selection

A feature vector for the emotional expression recognition should have all those essential features needed for classification. Unnecessary or irrelevant features can cause over-fitting due to the curse of dimensionality, as well as long learning and classification time. Hence, feature selection method is suggested as a preprocessing step to address the problem (Kumar, 2009:217-227). A feature selection method to be used for the emotional expression recognition must be a supervised one and must work with numeric values of the histograms. Therefore, a new method of feature selection is introduced. It selects a feature based on its power in discriminating the emotional expression classes. The discriminating power is measured by the difference between two variances of the feature value as follows. One is the variance of the feature for all given images, $VAR_a$, and the other is the average within-class variance of the feature value, $VAR_b$.

$$VAR_a = \frac{1}{N}\sum_{i=1}^{C}\sum_{j=1}^{N_i}(a_i^j - \bar{a})(a_i^j - \bar{a}) \tag{3.19}$$

$$VAR_b = \frac{1}{N}\sum_{i=1}^{C}\left(\frac{1}{N_i}\sum_{j=1}^{N_i}(a_i^j - \mu_i)(a_i^j - \mu_i)\right) * N_i \tag{3.20}$$

$$\Delta VAR = VAR_a - VAR_b \tag{3.21}$$

Where, $\bar{a} = \frac{1}{N}\sum_{i=1}^{C}\sum_{j=1}^{N_i} a_i^j$ and $\mu_i = \frac{1}{N_i}\sum_{j=1}^{N_i} a_i^j$

$a_i^j$ denotes the feature value of j-th training sample of the i-th emotional expression class, $\mu_i$ stands for the mean of the feature value of the i-th emotional expression class and $\bar{a}$ represents the mean of the feature value of all the training samples. $N_i$ is the number of training samples in the i-th class, *N* is the total number of training samples. $\Delta VAR$ represents the difference between the two variances. The high value of the variance difference for a feature means that the average within-class variance of the feature values is quite smaller than the total variance of the feature value of all the training samples regardless of their classes. Hence, the feature should be suitable for distinguishing samples from one class to the others, and so possesses high discriminating power. Features can then be ranked based on values of their $\Delta VAR$. A number of top ranked features can be selected and used for training and classification without degrading the accuracy of the classification while the feature length becomes smaller and requires less processing time. Experiments discussed in the next chapter will show how the selection would affect the performance of the classification system.

# CHAPTER 4

# EXPERIMENTS AND RESULTS

The extended Cohn-Kanade dataset (CK+) by Lucey et al. (2010: 94-101) and the Japanese Female Facial Expression (JAFFE) dataset (Kamachi, Lyons and Gyoba, 1998) are used for experiments to evaluate the effectiveness of the proposed method.

## 4.1 Extended Cohn-Kanade Dataset (CK+)

In CK+, there are 326 peak facial expressions from 123 subjects. Seven emotion categories are there. They are ‘Anger’, ‘Contempt’, ‘Disgust’, ‘Fear’, ‘Happy’, ‘Sadness’ and ‘Surprise’.

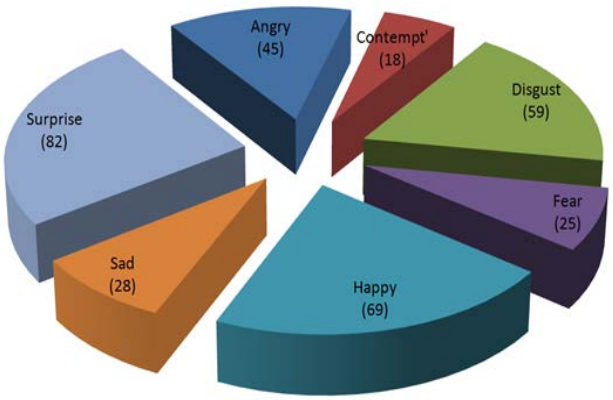



**Figure 4.1** CK+ Dataset, 7 Expressions and Number of Instances of Each Expression

Figure 4.1 shows the numbers of instances for each expression in the CK+ dataset. No subject with the same emotion has been collected more than once. All the facial images in the dataset are posed. Figure 4.2 shows some samples of facial expressions from the dataset.

**Figure 4.2** Some Samples from Cohn-Kanade (CK+) Dataset

## 4.2 Japanese Female Facial Expression Dataset (JAFFE)

JAFFE dataset contains 213 images of seven facial expressions (6 basic facial expressions + 1 neutral), posed by 10 Japanese female women. The dataset was planned and assembled by Miyuki Kamachi, Michael Lyons, and Jiro Gyoba in 1998. The photos were taken in the psychology department at Kyushu University.


**Figure 4.3** JAFFE Dataset, 7 Expressions and Number of Instances of Each Class

In the JAFFE dataset, same expression from the same subject was collected more than once. All the faces are posed. Number of each expression from all ten subjects is shown in Figure 4.3. Figure 4.4 shows some sample faces from JAFFE

dataset.

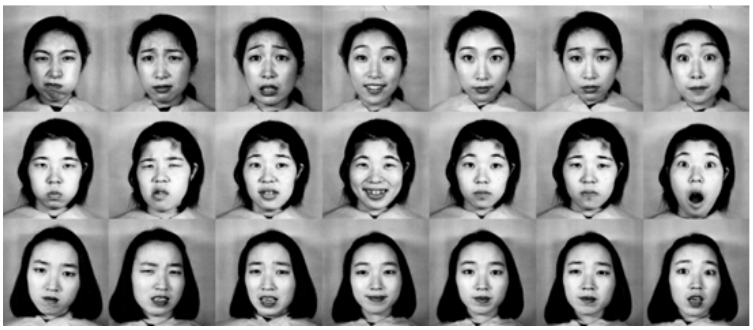


**Figure 4.4** Sample Faces from JAFFE Dataset

Table 4.1 shows the numbers of instances of expressions of both datasets.

**Table 4.1** Expression Instances from Each Dataset

| Expression Class | CK+ | JAFFE |
|---|---|---|
| 'Anger' | 45 | 30 |
| 'Disgust' | 59 | 29 |
| 'Fear' | 25 | 32 |
| 'Happy' | 69 | 31 |
| 'Sadness' | 28 | 31 |
| 'Surprise' | 82 | 30 |
| 'Contempt' | 18 | - |
| 'Neutral' | - | 30 |

## 4.3 Experiments

Face detection is done using *fdlibmex* library, free code available for Matlab. The library consists of single *mex* file with a single function that takes an image as input and returns the frontal face. The square sized face is normalised to a fixed dimension e.g. 180x180 for CK+ dataset and 99x99 for JAFFE dataset, which is near to their original dimension. The face image is then masked using an elliptical shape to extract only the face part. The masked image is divided into a number of equal sized blocks, which to some extend holds the features location information. Histogram of

GDP patterns calculated from each block is concatenated to build the feature vector.

The core of a facial expression recognition system is its classifier. A library for multiclass SVM known as LIBSVM (Chang and Lin, 2011) is used in proposed FER system as a classifier. Basic mechanism of Support Vector Machine is already explained earlier in chapter 1. Support Vector Machine is one of the most popular and a well-developed method in machine learning for classification, so in this work it is used mainly as a benchmark. Main features of LIBSVM include:

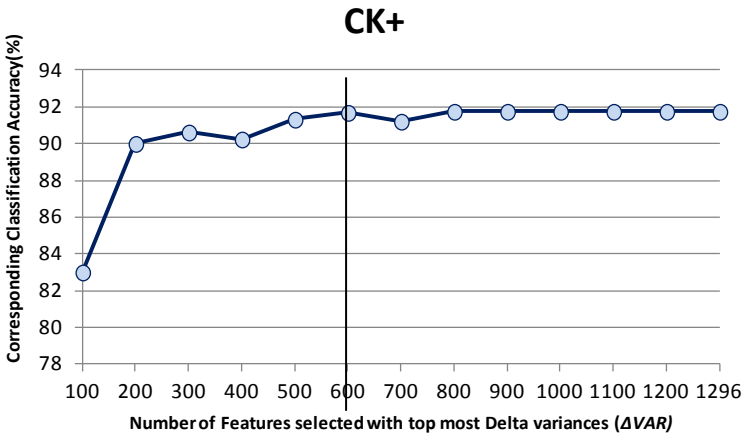
1) Different SVM formulations,
2) Effective multi-class classification,
3) Cross validation for class selection,
4) Probability estimations,
5) Varity of kernels (including pre computed kernel matrix),
6) Weighted SVM for unbalanced dataset,
7) Automatic model selection, which can generate contour of cross validation accuracy and so on.

A typical use of LIBSVM involves two steps: first, training a data set to obtain a model and second, using the model to predict information of a testing data set. For SVC (support vector classification) and SVR (support vector regression), LIBSVM can also output probability estimates. SVM formulations supported in LIBSVM: C-support vector classification (C-SVC), v-support vector classification (v-SVC), distribution estimation (one-class SVM), v-support vector regression (v-SVR), and v-support vector regression (v-SVR). The kernel parameters for the classifier are set to: s=0 for SVM type C-Svc, t=0/1/2 for linear, polynomial and RBF kernel function respectively, c=1 is the cost of SVM, g= 1/ (length of feature vector) and b=1 for probability estimation, see Appendix A. LIBSVM gives the result of the performance using confusion matrix and classification accuracy.

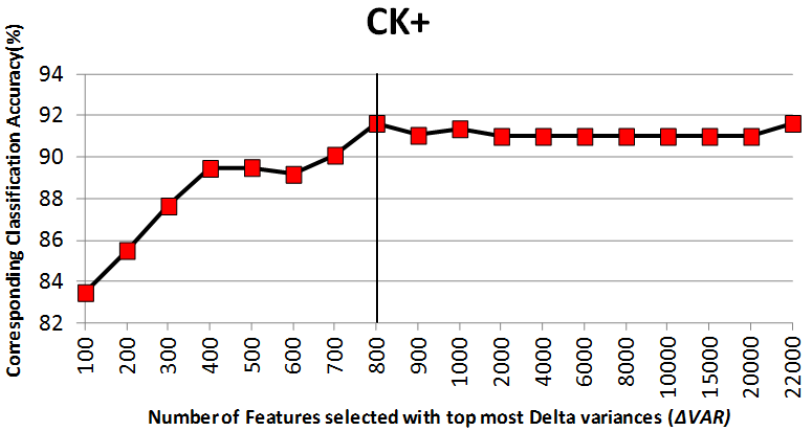
A ten-fold none overlapping cross validation was performed. The 90% of the images from each expression were used for training LIBSVM. The remaining 10 % of the images were used for testing. For each fold, different 10% of the images were chosen for testing and it is user-dependent. Ten rounds of training and testing were performed and the average confusion matrix and the average classification accuracy for the proposed method were reported.

### 4.3.1 First Set of Experiments

This set of experiments is intended to investigate the effects of the proposed feature selection method. The feature selection method mentioned in the previous chapter was performed on the two datasets. After features of all images in each dataset are derived, their respective ΔVAR values are computed, then the features are ranked by their ΔVAR. A number of top ranked features were selected to be used for training and classification. Several rounds of experiments were conducted with different numbers of selected features in order to find the optimal number of selected features that produce the best accuracy. The plotted graphs between the number of selected features vs. classification accuracy achieved from using GDP-4 and GDP-12 as its feature descriptor for both the datasets are shown in Figure 4.5.

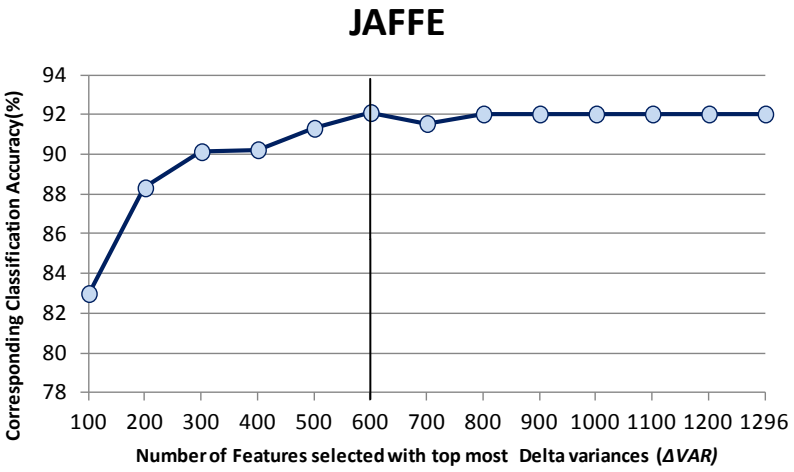



(a) For GDP-4 on CK+ dataset

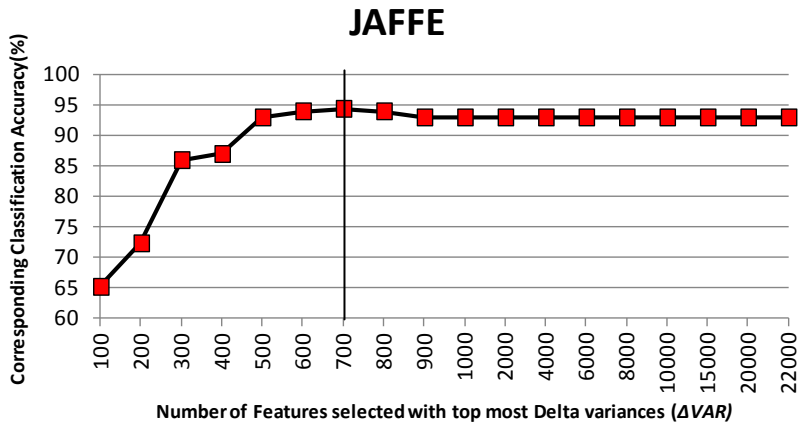



(b) For GDP-12 on CK+ dataset

**Figure 4.5** Plotted Graphs for Classification Accuracy vs. Number of Features Selected Using Top Ranked Δ***VAR***




(c) For GDP-4 on JAFFE dataset




(d) For GDP-12 on JAFFE dataset

**Figure 4.5** (Continued)

In case of GDP-4, feature vector length is 1296. Using ΔVAR as a measurement for feature selection, the feature vector can be reduced without hampering the recognition rate. Number of selected features vs. classification accuracy on CK+ and JAFFE datasets is shown in Figure 4.6 (a) and Figure 4.6 (c) respectively. It can be seen from Figure 4.6 (a) and (c) that the number of features can be reduced about half without sacrificing the accuracy using ΔVAR for feature selection. It is clear from those two graphs that number of top ranked features between 600-700 gives as good accuracy as the full features.

In case of GDP-12, feature vector length is 22032. The length can slow down the training and classification procedures significantly. Using ΔVAR for feature selection, the feature vector can be reduced without hampering the recognition rate. Number of selected features vs. classification accuracy on CK+ and JAFFE datasets is shown in Figure 4.6 (b) and Figure 4.6 (d) respectively. It shows that only 700 features selected by ΔVAR procedure give the same recognition results as for the full feature selection. So the ΔVAR procedure cuts the feature vector length by nearly 32 times.

By examining the GDP patterns of the features selected using ΔVAR on both datasets, the patterns are found to consist of at most one 0-1 or 1-0 transition, i.e. for GDP-4, 0000, 0001, 0011, 0111, 1111, 1000, 1100 and 1110. Therefore, the features can be selected if their corresponding GDP patterns consist at most one 0-1 or 1-0 transitions. This makes the selected feature patterns uniform and easy to check but does not affect the accuracy significantly.

The classification accuracy results achieved before and after feature selection for both datasets are shown in Table 4.2. It can be seen from the results that the feature selection does not affect the accuracy achieved on both datasets while helps reduce the number of features significantly. It can also be seen that both selection methods, one uses ΔVAR for selection and the other selects only the features with uniform patterns, give quite comparable accuracy of classification.

Table 4.3 shows the comparison of feature lengths per block for the proposed methods before and after feature selection (both selection methods) as well as some other well-known methods.

**Table 4.2** Classification Accuracy before and after Feature Dimension Reduction

| Dataset | Feature selection | | Classification Accuracy | | | |
|---|---|---|---|---|---|---|
| | | | GDP-2a | GDP-2b | GDP-4 | GDP-12 |
| CK+ | Before | | N/A | N/A | 91.75% | 91.63% |
| | After | ΔVAR | | | 91.69% | 91.63% |
| | | Uniform | | | 91.69% | 91.63% |
| JAFFE | Before | | N/A | N/A | 92.04% | 92.94% |
| | After | ΔVAR | | | 92.11% | 94.41% |
| | | Uniform | | | 92.11% | 92.94% |

**Table 4.3** Comparison of Feature Lengths per Block for the Proposed Methods before and after Feature Selection as well as Some Other Well-known Methods

| Feature selection | | Feature length per block | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | GDP-2a | GDP-2b | GDP-4 | GDP-12 | LBP | $LBP_{U2}$ | LPQ |
| Before | | 4 | 4 | 16 | 272 | 256 | 59 | 256 |
| After | ΔVAR | N/A | N/A | 8 | 10 | N/A | N/A | N/A |
| | Uniform | | | 8 | 24 | 59($LBP_{U2}$) | N/A | N/A |

### 4.3.2 Second Set of Experiments

Several experiments were conducted on CK+ datasets to compare the performances of the proposed methods with uniform pattern features and other well-known methods, LBP and LPQ. Several block numbers are also tried to see the effect of the block numbers with the accuracy. The classification accuracy vs. the block numbers using GDP, LBP and LPQ are given in Table 4.4. The number of blocks of 9x9=81 is found to be the best among all the combinations.

**Table 4.4** Block Dimension vs. Classification Accuracy (CK+ dataset)

| **Blocks** | **Classification Accuracy (%)** | | | | | | |
|---|---|---|---|---|---|---|---|
| | GDP-2a | GDP-2b | GDP-4 | GDP-12 | LBP | $LBP_{U2}$ | LPQ |
| **10x10** | 87.72% | 88.10% | 91.21% | 91.52% | 90.54% | 89.65% | 80.21% |
| 9x9 | **87.70%** | **88.00%** | **91.69%** | **91.63%** | **90.11%** | **90.12%** | **80.21%** |
| **9x8** | 86.82% | 87.12% | 90.77% | 90.71% | 89.21% | 89.22% | 79.41% |
| **8x9** | 85.95% | 86.25% | 89.87% | 89.81% | 88.32% | 88.33% | 78.61% |
| **8x8** | 85.10% | 85.39% | 88.97% | 88.91% | 88.65% | 87.44% | 80.32% |
| **7x8** | 84.24% | 86.21% | 89.24% | 89.32% | 87.76% | 88.37% | 79.52% |
| **8x7** | 84.40% | 85.35% | 88.35% | 88.43% | 86.89% | 87.49% | 78.72% |
| **7x7** | 83.56% | 84.49% | 87.46% | 87.54% | 86.02% | 86.61% | 77.93% |
| **6x7** | 82.72% | 83.65% | 86.59% | 86.67% | 85.16% | 85.75% | 77.16% |

The average of ten-fold cross validation for facial expression recognition using the proposed feature representation method on CK+ dataset are shown using confusion matrices in Table 4.5.

**Table 4.5** Confusion Matrices Results for CK+ Dataset

| prediction \ Actual | Angry | Contempt | Disgust | Fear | Happy | Sad | Surprise |
|---|---|---|---|---|---|---|---|
| Angry | **77.8** | 4.4 | 8.9 | 0.0 | 0.0 | 8.9 | 0.0 |
| Contempt | 16.7 | **72.2** | 0.0 | 5.6 | 0.0 | 0.0 | 5.6 |
| Disgust | 6.8 | 0.0 | **91.5** | 1.7 | 0.0 | 0.0 | 0.0 |
| Fear | 4.0 | 8.0 | 4.0 | **60.0** | 12.0 | 0.0 | 12.0 |
| Happy | 0.0 | 0.0 | 0.0 | 0.0 | **100.0** | 0.0 | 0.0 |
| Sad | 14.3 | 3.6 | 10.7 | 3.6 | 0.0 | **67.9** | 0.0 |
| Surprise | 1.2 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | **98.8** |

a) GDP-2a

| prediction \ Actual | Angry | Contempt | Disgust | Fear | Happy | Sad | Surprise |
|---|---|---|---|---|---|---|---|
| Angry | **73.3** | 6.7 | 8.9 | 2.2 | 0.0 | 8.9 | 0.0 |
| Contempt | 16.7 | **77.8** | 0.0 | 0.0 | 0.0 | 0.0 | 5.6 |
| Disgust | 0.0 | 0.0 | **94.9** | 3.4 | 0.0 | 1.7 | 0.0 |
| Fear | 4.0 | 8.0 | 4.0 | **76.0** | 8.0 | 0.0 | 0.0 |
| Happy | 0.0 | 0.0 | 0.0 | 0.0 | **98.6** | 1.4 | 0.0 |
| Sad | 28.6 | 7.1 | 3.6 | 0.0 | 0.0 | **60.7** | 0.0 |
| Surprise | 0.0 | 0.0 | 1.2 | 0.0 | 0.0 | 1.2 | **97.6** |

b) GDP-2b

| prediction \ Actual | Angry | Contempt | Disgust | Fear | Happy | Sad | Surprise |
|---|---|---|---|---|---|---|---|
| Angry | **82.2** | 6.7 | 4.4 | 0.0 | 0.0 | 6.7 | 0.0 |
| Contempt | 11.1 | **77.8** | 0.0 | 0.0 | 0.0 | 11.1 | 0.0 |
| Disgust | 1.7 | 0.0 | **94.9** | 1.7 | 1.7 | 0.0 | 0.0 |
| Fear | 4.0 | 4.0 | 4.0 | **84.0** | 4.0 | 0.0 | 0.0 |
| Happy | 0.0 | 0.0 | 0.0 | 0.0 | **100.0** | 0.0 | 0.0 |
| Sad | 14.3 | 0.0 | 0.0 | 7.1 | 0.0 | **75.0** | 3.6 |
| Surprise | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | **100.0** |

c) GDP-4

| prediction \ Actual | Angry | Contempt | Disgust | Fear | Happy | Sad | Surprise |
|---|---|---|---|---|---|---|---|
| Angry | **82.2** | 4.4 | 8.9 | 2.2 | 0.0 | 2.2 | 0.0 |
| Contempt | 16.7 | **72.2** | 0.0 | 0.0 | 0.0 | 11.1 | 0.0 |
| Disgust | 1.7 | 0.0 | **96.6** | 1.7 | 0.0 | 0.0 | 0.0 |
| Fear | 4.0 | 4.0 | 4.0 | **84.0** | 4.0 | 0.0 | 0.0 |
| Happy | 0.0 | 0.0 | 0.0 | 0.0 | **100.0** | 0.0 | 0.0 |
| Sad | 14.3 | 0.0 | 0.0 | 3.6 | 0.0 | **78.6** | 3.6 |
| Surprise | 0.0 | 1.2 | 0.0 | 1.2 | 0.0 | 0.0 | **97.6** |

d) GDP-12

| prediction \ Actual | Angry | Contempt | Disgust | Fear | Happy | Sad | Surprise |
|---|---|---|---|---|---|---|---|
| Angry | **77.8** | 4.4 | 6.7 | 2.2 | 0.0 | 8.9 | 0.0 |
| Contempt | 11.1 | **83.3** | 0.0 | 0.0 | 0.0 | 5.6 | 0.0 |
| Disgust | 1.7 | 0.0 | **96.6** | 1.7 | 0.0 | 0.0 | 0.0 |
| Fear | 8.0 | 4.0 | 4.0 | **72.0** | 8.0 | 0.0 | 4.0 |
| Happy | 1.4 | 0.0 | 0.0 | 1.4 | **97.1** | 0.0 | 0.0 |
| Sad | 10.7 | 3.6 | 0.0 | 7.1 | 0.0 | **78.6** | 0.0 |
| Surprise | 1.2 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | **98.8** |

e) LBP

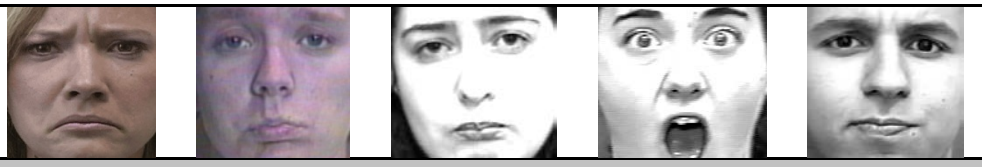
| prediction \ Actual | Angry | Contempt | Disgust | Fear | Happy | Sad | Surprise |
|---|---|---|---|---|---|---|---|
| Angry | **71.1** | 6.7 | 13.3 | 2.2 | 0.0 | 6.7 | 0.0 |
| Contempt | 11.1 | **83.3** | 0.0 | 5.6 | 0.0 | 0.0 | 0.0 |
| Disgust | 1.7 | 0.0 | **96.6** | 1.7 | 0.0 | 0.0 | 0.0 |
| Fear | 8.0 | 4.0 | 4.0 | **72.0** | 12.0 | 0.0 | 0.0 |
| Happy | 0.0 | 1.4 | 0.0 | 0.0 | **98.6** | 0.0 | 0.0 |
| Sad | 7.1 | 7.1 | 3.6 | 3.6 | 0.0 | **78.6** | 0.0 |
| Surprise | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | **100.0** |

f) $LBP_{U2}$

| prediction \ Actual | Angry | Contempt | Disgust | Fear | Happy | Sad | Surprise |
|---|---|---|---|---|---|---|---|
| Angry | **62.2** | 6.7 | 11.1 | 2.2 | 2.2 | 13.3 | 2.2 |
| Contempt | 16.7 | **66.7** | 5.6 | 0.0 | 0.0 | 5.6 | 5.6 |
| Disgust | 11.9 | 0.0 | **78.0** | 3.4 | 5.1 | 0.0 | 1.7 |
| Fear | 12.0 | 8.0 | 8.0 | **52.0** | 8.0 | 4.0 | 8.0 |
| Happy | 0.0 | 0.0 | 2.9 | 1.4 | **92.8** | 0.0 | 2.9 |
| Sad | 35.7 | 7.1 | 0.0 | 0.0 | 0.0 | **42.9** | 14.3 |
| Surprise | 2.4 | 0.0 | 1.2 | 0.0 | 0.0 | 0.0 | **96.3** |

g) LPQ

It can be seen from the confusion matrices that some particular expression classes, e.g. contempt and fear, are consistently more difficult to classify than the others. Some instances of these expressions are consistently misclassified when using the GDP. These instances are difficult to distinguish even by a human, see the instances in Figure 4.6.



| | | | | | |
|---|---|---|---|---|---|
| **Original Class** | Angry | Contempt | Sad | Surprise | Angry |
| **Classified Class** | Sad | Sad | Contempt | Angry | Contempt |

**Figure 4.6** Some Instances of Consistently Misclassified Expressions when Using the GDP

The achieved classification accuracy, feature extraction time for a single facial image, learning time for a single fold and classification time of a single image are shown in comparison with those of the proposed method in Table 4.6.

**Table 4.6** Classification Accuracy and Processing Time Comparison for CK+ Dataset

| | CK+ | | | |
|---|---|---|---|---|
| **Method** | **Classification Accuracy (%)** | **Feature Extraction Time** | **Learning Time** | **Classification Time** |
| GDP-2a | 87.70% | 0.011 sec | 0.901 sec | 0.001 sec |
| GDP-2b | 88.00% | 0.011 sec | 0.910 sec | 0.001 sec |
| GDP-4 | 91.96% | 0.019 sec | 1.660 sec | 0.002 sec |
| GDP-12 | 91.63% | 0.08 sec | 6.110 sec | 0.009 sec |
| LPQ | 80.21% | 0.29 sec | 66.00 sec | 0.070 sec |
| LBP | 90.11% | 0.07 sec | 66.00 sec | 0.070 sec |
| $LBP_{u2}$ | 90.12% | 0.06 sec | 37.00 sec | 0.025 sec |

Table 4.7 compares the accuracy achieved by the proposed GDP-12 method on the CK+ dataset with those of other recent methods. It should be noted that although the results shown in the table came from the experiments with different experimental setups, different versions of the CK, different preprocessing methods, and so on, but they still point out the discriminative power of each method. The execution time of all the methods cannot be compared due to differences on experimental setup and execution environments.

**Table 4.7** Comparison of Classification Accuracy Achieved by GDP-12 Method with Those of Some Other Recent Methods on CK+ Dataset

| | Method | No of subjects | No of Images | Classification accuracy (%) |
|---|---|---|---|---|
| Chew et al. (2011: 915-920) | Appearance-based (PDM) | 123 | 327 | 80+% |
| Naika et al. (2012: 244-252) | Appearance-based (EAR-LBP) | 123 | 327 | 82% |
| Yang & Bhanu (2012: 980-992) | Appearance-based (LBP + LPQ) | 123 | 316 | 83% |
| Jeni et al. (2012: 785-795) | Shape-based (68 Landmarks) | 123 | 593 | 87% |
| **Proposed** | **GDP-12** | **123** | **326** | **92%** |

The same experimental setup is followed for JAFFE dataset as in the CK+, except the face dimension. The face dimension in this case is 99x99 pixels. This is because, the images in JAFFE are of 256x256 pixels in dimension, which is nearly half of the CK+ images (Figure 4.7). The original face dimension detected by face detector is little more or less than 99x99 pixels.

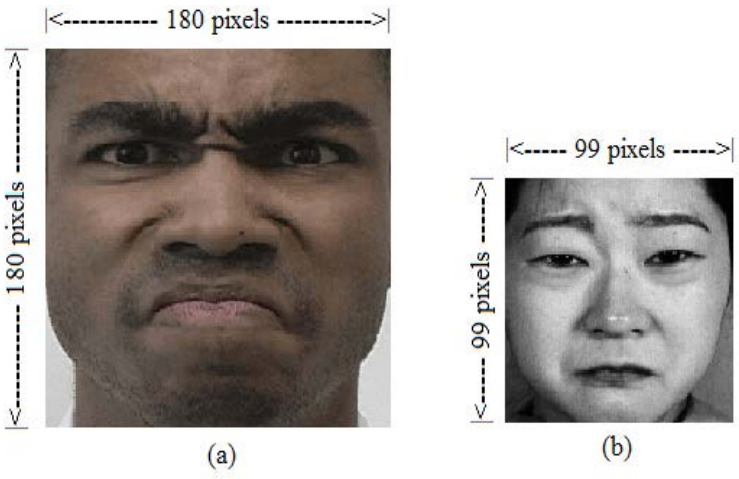



**Figure 4.7** Normalized Facial Sample from (a) CK+ Dataset and (b) JAFFE Dataset

Table 4.8 shows the number of blocks that yields the best performance in terms of accuracy for JAFFE dataset.

**Table 4.8** Block Dimension vs. Classification Accuracy (JAFFE dataset)

| Blocks | Classification Accuracy (%) | | | | | | |
|---|---|---|---|---|---|---|---|
| | **GDP-2a** | **GDP-2b** | **GDP-4** | **GDP-12** | **LBP** | **$LBP_{U2}$** | **LPQ** |
| 10x10 | 86.10% | 88.60% | 92.00% | 92.51% | 90.98% | 91.16% | 79.90% |
| **9x9** | **86.30%** | **88.60%** | **92.11%** | **92.94%** | **90.98%** | **91.14%** | **79.60%** |
| 9x8 | 85.21% | 86.65% | 91.01% | 91.21% | 90.64% | 91.12% | 79.56% |
| 8x9 | 85.12% | 87.63% | 90.21% | 88.23% | 90.88% | 91.00% | 78.36% |
| 8x8 | 85.00% | 87.21% | 91.32% | 89.21% | 88.56% | 89.54% | 78.78% |
| 7x8 | 84.51% | 85.63% | 89.20% | 90.78% | 90.47% | 90.86% | 78.21% |
| 8x7 | 83.56% | 87.65% | 88.36% | 88.96% | 90.78% | 91.01% | 78.42% |
| 7x7 | 84.32% | 85.32% | 90.21% | 91.10% | 88.65% | 89.54% | 78.35% |
| 6x7 | 83.95% | 86.21% | 89.17% | 88.23% | 89.47% | 88.15% | 76.48% |

The number of blocks of 9x9 or 81 yields the best accuracy. Due to smaller face dimension for JAFFE dataset, each of the 81 block is of size 11x11 pixels. Unlike CK+ dataset, in JAFFE single subject has more instances of the same expression, e.g. 2-4 times. Therefore, average expressions from a single subject are 21 for seven classes of expression. The results obtained from the proposed facial expression recognition system using GDP-2a, GDP-2b, GDP-4, GDP-12 and some other popular methods are shown in Table 4.9 using confusion matrices. Figure 4.8 shows some instances of the dataset that are consistently misclassified using the GDP.

**Table 4.9** Confusion Matrices Results for JAFFE Dataset

Actual

| prediction | Angry | Disgust | Fear | Happy | Neutral | Sad | Surprise |
|---|---|---|---|---|---|---|---|
| Angry | **96.7** | 3.3 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Disgust | 6.9 | **86.2** | 3.4 | 0.0 | 3.4 | 0.0 | 0.0 |
| Fear | 0.0 | 6.3 | **59.4** | 3.1 | 15.6 | 9.4 | 6.3 |
| Happy | 0.0 | 0.0 | 0.0 | **93.5** | 6.5 | 0.0 | 0.0 |
| Neutral | 0.0 | 0.0 | 0.0 | 0.0 | **93.3** | 6.7 | 0.0 |
| Sad | 3.2 | 0.0 | 9.7 | 3.2 | 0.0 | **83.9** | 0.0 |
| Surprise | 0.0 | 0.0 | 3.3 | 3.3 | 6.7 | 0.0 | **86.7** |

a) GDP-2a

Actual

| prediction | Angry | Disgust | Fear | Happy | Neutral | Sad | Surprise |
|---|---|---|---|---|---|---|---|
| Angry | **96.7** | 0.0 | 0.0 | 0.0 | 3.3 | 0.0 | 0.0 |
| Disgust | 0.0 | **89.7** | 6.9 | 0.0 | 0.0 | 3.4 | 0.0 |
| Fear | 0.0 | 3.1 | **75.0** | 6.3 | 3.1 | 9.4 | 3.1 |
| Happy | 0.0 | 0.0 | 0.0 | **87.1** | 3.2 | 6.5 | 3.2 |
| Neutral | 0.0 | 0.0 | 0.0 | 0.0 | **100.0** | 0.0 | 0.0 |
| Sad | 0.0 | 0.0 | 6.5 | 3.2 | 6.5 | **83.9** | 0.0 |
| Surprise | 0.0 | 0.0 | 0.0 | 6.7 | 3.3 | 0.0 | **90.0** |

b) GDP-2b

Actual

| prediction | Angry | Disgust | Fear | Happy | Neutral | Sad | Surprise |
|---|---|---|---|---|---|---|---|
| Angry | **93.3** | 0.0 | 0.0 | 0.0 | 0.0 | 6.7 | 0.0 |
| Disgust | 6.9 | **86.2** | 3.4 | 0.0 | 0.0 | 3.4 | 0.0 |
| Fear | 0.0 | 3.1 | **81.3** | 3.1 | 3.1 | 6.3 | 3.1 |
| Happy | 0.0 | 0.0 | 0.0 | **96.8** | 0.0 | 3.2 | 0.0 |
| Neutral | 0.0 | 0.0 | 0.0 | 0.0 | **96.7** | 3.3 | 0.0 |
| Sad | 3.2 | 0.0 | 3.2 | 3.2 | 0.0 | **90.3** | 0.0 |
| Surprise | 0.0 | 0.0 | 3.3 | 3.3 | 0.0 | 0.0 | **93.3** |

c) GDP-4

Actual

| prediction | Angry | Disgust | Fear | Happy | Neutral | Sad | Surprise |
|---|---|---|---|---|---|---|---|
| Angry | **93.3** | 0.0 | 0.0 | 0.0 | 0.0 | 6.7 | 0.0 |
| Disgust | 3.4 | **89.7** | 6.9 | 0.0 | 0.0 | 0.0 | 0.0 |
| Fear | 0.0 | 3.1 | **84.4** | 3.1 | 0.0 | 6.3 | 3.1 |
| Happy | 0.0 | 0.0 | 0.0 | **96.8** | 0.0 | 3.2 | 0.0 |
| Neutral | 0.0 | 0.0 | 0.0 | 0.0 | **100.0** | 0.0 | 0.0 |
| Sad | 0.0 | 0.0 | 9.7 | 3.2 | 0.0 | **87.1** | 0.0 |
| Surprise | 0.0 | 0.0 | 0.0 | 3.3 | 3.3 | 0.0 | **93.3** |

d) GDP-12

**Table 4.9** (continued)

| prediction \ Actual | Angry | Disgust | Fear | Happy | Neutral | Sad | Surprise |
|---|---|---|---|---|---|---|---|
| Angry | **86.7** | 0.0 | 0.0 | 0.0 | 6.7 | 6.7 | 0.0 |
| Disgust | 10.3 | **82.8** | 3.4 | 0.0 | 0.0 | 3.4 | 0.0 |
| Fear | 0.0 | 3.1 | **87.5** | 0.0 | 3.1 | 3.1 | 3.1 |
| Happy | 0.0 | 0.0 | 0.0 | **96.8** | 0.0 | 3.2 | 0.0 |
| Neutral | 0.0 | 0.0 | 0.0 | 0.0 | **100.0** | 0.0 | 0.0 |
| Sad | 0.0 | 0.0 | 6.5 | 3.2 | 0.0 | **90.3** | 0.0 |
| Surprise | 0.0 | 0.0 | 0.0 | 3.3 | 6.7 | 0.0 | **90.0** |

e) LBP

| prediction \ Actual | Angry | Disgust | Fear | Happy | Neutral | Sad | Surprise |
|---|---|---|---|---|---|---|---|
| Angry | **86.7** | 0.0 | 0.0 | 0.0 | 6.7 | 6.7 | 0.0 |
| Disgust | 10.3 | **86.2** | 0.0 | 0.0 | 0.0 | 3.4 | 0.0 |
| Fear | 0.0 | 3.1 | **84.4** | 0.0 | 6.3 | 3.1 | 3.1 |
| Happy | 0.0 | 0.0 | 0.0 | **100.0** | 0.0 | 0.0 | 0.0 |
| Neutral | 0.0 | 0.0 | 0.0 | 0.0 | **100.0** | 0.0 | 0.0 |
| Sad | 0.0 | 0.0 | 6.5 | 3.2 | 0.0 | **87.1** | 3.2 |
| Surprise | 0.0 | 0.0 | 0.0 | 3.3 | 6.7 | 0.0 | **90.0** |

f) $LBP_{U2}$

| prediction \ Actual | Angry | Disgust | Fear | Happy | Neutral | Sad | Surprise |
|---|---|---|---|---|---|---|---|
| Angry | **83.3** | 0.0 | 3.3 | 10.0 | 0.0 | 3.3 | 0.0 |
| Disgust | 3.4 | **86.2** | 10.3 | 0.0 | 0.0 | 0.0 | 0.0 |
| Fear | 0.0 | 9.4 | **68.8** | 6.3 | 6.3 | 6.3 | 3.1 |
| Happy | 0.0 | 9.7 | 0.0 | **74.2** | 9.7 | 6.5 | 0.0 |
| Neutral | 0.0 | 0.0 | 3.3 | 0.0 | **86.7** | 10.0 | 0.0 |
| Sad | 0.0 | 0.0 | 12.9 | 3.2 | 3.2 | **80.6** | 0.0 |
| Surprise | 0.0 | 0.0 | 10.0 | 3.3 | 3.3 | 0.0 | **83.3** |

g) LPQ

| **Original Class** | Angry | Disgust | Fear | Surprise | Sad |
|---|---|---|---|---|---|
| **Classified Class** | Neutral | Angry | Neutral | Happy | Happy |

**Figure 4.8** Some Instances of Consistently Misclassified Expressions when Using the GDP from JAFFE Dataset

**Table 4.10** Classification Accuracy and Processing Time Comparison for JAFFE Dataset

| | JAFFE | | | |
|---|---|---|---|---|
| **Method** | **Classification Accuracy (%)** | **Feature Extraction Time** | **Learning Time** | **Classification Time** |
| GDP-2a | 86.30% | 0.004 sec | 0.609 sec | 0.001 sec |
| GDP-2b | 88.60% | 0.004 sec | 0.704 sec | 0.001 sec |
| GDP-4 | 92.11% | 0.006 sec | 1.160 sec | 0.002 sec |
| GDP-12 | 92.94% | 0.024 sec | 4.101 sec | 0.009 sec |
| LPQ | 79.60% | 0.095 sec | 46.00 sec | 0.070 sec |
| LBP | 91.14% | 0.026 sec | 46.00 sec | 0.070 sec |
| $LBP_{u2}$ | 90.98% | 0.025 sec | 25.00 sec | 0.025 sec |

The achieved classification accuracy, feature extraction time for a single facial image, learning time for a single fold and classification time of a single image are shown in comparison in Table 4.10. Table 4.11 compares the classification accuracy achieved by GDP-12 with those of recent methods on JAFFE dataset.

**Table 4.11** Comparison of Classification Accuracy Achieved by GDP-12 with those of Some Other Recent Methods on JAFFE Dataset

(NN: Neural Network, LDA: Local Discriminant Analysis)

| Author | Method | Classifier | Classification Accuracy |
|---|---|---|---|
| **Proposed** | **GDP-12** | **Multi Class SVM (Poly)** | **92.94%** |
| Subramanian et al. (2012: 1-7) | LBP | SVM | 88.09% |
| Lyons et al. * (1999: 1357-1362) | Gabor Filter | LDA-based classification | 92.00% |
| Zhang et al. (1998: 454-459) | Gabor Filter | NN | 90.10% |
| Guo & Dyer, 2003 | Gabor Filter | Linear Programming | 91.00% |

**Note:** *Used a Subset of the Dataset

It should be noted that the results shown in the table came from the experiments with different experimental setups, different classification techniques, different preprocessing methods, and so on, but they still point out the discriminative power of each method.

The execution time of all the methods can not be compared due to differences on experimental setup and execution environments. It can also be shown from the experimental results of the two datasets that the GDP-12, which considers the most numbers of considered directions and neighboring pixels, achieves the best performance in term of accuracy. This may be due the richest information the local pattern possesses. However, its pattern length is the longest and so the longest processing time is needed. The GDP-4 achieve somewhat less accuracy than the GDP-12 but requires shorter pattern length and so less processing time. Hence, to achieve better accuracy for the GDP, longer feature length and processing time are needed.

# CHAPTER 5

# CONCLUSION AND FUTURE WORK

## 5.1 Conclusion

A framework for facial expression recognition is provided in this thesis. Mainly two issues are discussed in details, (a) Facial feature extraction and (b) Feature selection. Both the issues are challenging problem, and significant research effort has been given towards finding the appropriate solutions for them.

For each pixel in a gray scale image, the proposed feature representation method, namely gradient direction pattern (GDP) extracts the local binary pattern using gradient directions between two opposite neighboring pixels in 3x3 or 5x5 region. The pattern represents the changes on the gray color values of pixels in its surrounding area and so the unique local feature for the considered pixel. The pattern is also invariant to the light condition. Four possible GDP extraction are proposed depending on the numbers of considered gradient directions and the size of the neighborhood regions. The more the numbers, the richer the pattern represents and so the higher classification accuracy will be achieved from using it. However, as the results, more memory space and processing time will be needed.

A variance-based feature selection is proposed and used to reduce the number of features by half for GDP-4 and by 12 times for GDP-12. The resultant features become uniform with no more than one e.g. 0-1 or 1-0 transition in the binary patterns. Further selection from the uniform feature can be done in case of GDP-12 that reduces the feature vector length again by one-third. The GDP, especially, GDP-12 and GDP-4, are very effective for facial expression recognition and can outperform LPQ, LBP, and $LBP_{U2}$, in terms of both classification accuracy and processing time. The classification accuracy achieved by the GDP is also better than those achieved by some other recent works on facial expression recognition.

## 5.2 Major Contributions

In this thesis, the main contributions are:

1) A new appearance-based feature representation method for facial expression recognition is proposed. The method consists of four alternative feature representations, namely (a) GDP-2a, (b) GDP-2b, (c) GDP-4, and (d) GDP-12. The proposed feature representations are gray scale invariant, very effective for facial expression recognition, and easy to compute. These characteristics makes them suitable for real time applications.

2) A new method for facial feature selection is introduced based on feature variances. The method leads to the selection of only uniform patterns with single or less transition. The selected subset of the full features has been shown to be good enough to differentiate the facial expressions.

## 5.3 Limitations and Future Work

The proposed system along with multi-class support vector machine performs well for the tasks it is designed for. However, the system has some practical limitations as this is a research work. For example, all of the learning and recognition performed are far from real time environment.

The datasets used have posed images only. A possible future research direction is to incorporate variations on face pose, which will add more degrees of freedom to manifold of expressions.

In practical situation, a subject may speak and smile or give other facial expressions simultaneously. Therefore, the mouth region is affected by both expression and vocal content. In such situation, one needs to put and adjust some weight for upper-part and lower part of the face for exact facial expression recognition. There is no such data for talking face and expression face in both the datasets used in this work. Therefore, a more practical-system can be built by using datasets having natural expressions while talking.

# BIBLIOGRAPHY


Ahonen, T., Hadid, A., & Pietikäinen, M. (2006). Face description with local binary patterns: application to face recognition. *IEEE Transactions On Pattern Analysis and Machine Intelligence*, *28*(12), 2037–2041.

Ahsan, T., Jabid, T., & Chong, U. P. (2013). Facial expression recognition using local transitional pattern on gabor filtered facial images. *IETE Technical Review*, *30*(1), 47.

Aizerman, A., Braverman, E. M., & Rozoner, L. I. (1964). Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, *25*, 821-837.

Aleksic, P. S., & Katsaggelos, A. K. (2006). Facial animation parameters and multistream HMMs. *IEEE Transactions on Information Forensics and Security*, *1*(1), 3–11.

Bartlett, M. S., Littlewort, G., Fasel, I., & Movellan, J. R. (2003). Real time face detection and facial expression recognition: development and applications to human computer interaction. In *2003 Conference on Computer Vision and Pattern Recognition Workshop*: Vol. 5 (p. 53). doi:10.1109/CVPRW.2003.10057

Bartlett, M. S., Littlewort, G., Frank, M., Lainscsek, C., Fasel, I., & Movellan, J. (2005). Recognizing facial expression: machine learning and application to spontaneous behavior. In *2005 IEEE Computer Society Conference on Computer Vision & Pattern Recognition (CVPR'05)*: Vol. 2 (pp. 568–573). doi:10.1109/CVPR.2005.297

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory* (pp. 144–152). New York: ACM.

Chang, C. C., & Lin, C. J. (2011). LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *2*(3), 27.

Chew, S. W., Lucey, P., Lucey, S., Saragih, J., Cohn, J. F., & Sridharan, S. (2011). Person-independent facial expression detection using constrained local models. In *2011 IEEE International Conference on Automatic Face & Gesture Recognition and Workshops (FG 2011),* (pp. 915-920). Santabarbara, CA: IEEE.

Choi, S.-M., & Kim, Y.-G. (2005). An affective user interface based on facial expression recognition and eye-gaze tracking. In J. Tao, T. Tan, & R. Picard (Eds.). *Affective Computing and Intelligent Interaction SE – 116:* Vol. 3784 (pp. 907–914). Berlin: Springer.

Colmenarez, A., Frey, B., & Huang, T. S. (1999). A probabilistic framework for embedded face and facial expression recognition. In *Proceeding of IEEE Computer Society Conference on Computer Vision and Pattern Recognition:* Vol.2 (pp. 592-597). Los Alamos, CA: IEEE.

Cootes, T. F., Edwards, G. J., & Taylor, C. J. (2001). Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(6), 681–685.

Cristinacce, D., & Cootes, T. (2008). Automatic feature localisation with constrained local models. *Pattern Recognition*, *41*(10), 3054–3067.

Ekman, Paul. (2005). Basic emotions. In *Handbook of Cognition and Emotion* (pp. 45–60). doi:10.1002/0470013494.

Ekman, P., & Friesen, W. V. (1978). *Facial action coding system: A technique for the measurement of facial movement*. Palo Alto, CA: Consulting Psychologists Press.

Ekman, P., Friesen, W. V., & Hager, J. C. (2002). *Facial action coding system*. Salt Lake City, UT: A Human Face.

Ekman, P, Rosenberg, E., & Hager, J. (1998). Facial action coding system affect interpretation dictionary (FACSAID). Retrieved on July 2012 from http://face-and-emotion.com/dataface/facsaid/description.jsp

Ellsworth, P.C., & Smith, C.A. (1988). From appraisal to emotion: Differences among unpleasant feelings. *Motivation and Emotion*, *12*, 271-302.

Friesen, W. V., & Ekman, P. (1983). *Emfacs-7: emotional facial action coding system*. (Unpublished manuscript). Sanfransisco, CA: University of

California at San Francisco.

Guo, G., & Dyer, C. R. (2003). Simultaneous feature selection and classifier training via linear programming: A case study for face expression recognition. In *Proceeding of 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition:* Vol. 1. (pp. I-346). IEEE.

Hamm, J., Kohler, C. G., Gur, R. C., & Verma, R. (2011). Automated facial action coding system for dynamic analysis of facial expressions in neuropsychiatric disorders. *Journal Of Neuroscience Methods*, *200*(2), 237-256.

Har-Peled, S., Roth, D., & Zimak, D. (2002). Constraint Classification for Multiclass Classification and Ranking. *Advances in neural information processing systems*, *15*, 785-792.

Heisele, B., & Koshizen, T. (2004). Components for face recognition. In *Proceedings of the Sixth IEEE International Conference on Automatic Face and Gesture Recognition* (pp. 153-158). Seol, SK: IEEE.

Huang, X., Zhao, G., Pietikäinen, M., & Zheng, W. (2010). Dynamic facial expression recognition using boosted component-based spatiotemporal features and multi-classifier fusion. In *Advanced Concepts for Intelligent Vision Systems* (pp. 312-322). Berlin: Springer.

Huang, X., Zhao, G., Pietikäinen, M., & Zheng, W. (2011). Expression recognition in videos using a weighted component-based feature descriptor. In *Image Analysis* (pp. 569-578). Berlin: Springer.

Huber, E. (1931). Evolution of facial musculature and facial expression. Retrieved on July 2012, from http://psycnet.apa.org/psycinfo/1931-04729-000

Ichimura, T., Oeda, S., & Yamashita, T. (2002). Construction of emotional space from facial expression by parallel sand glass type neural networks. In *Proceedings of the International Joint Conference on Neural Networks:* (pp. 2422–2427). The United States: IEEE

Jabid, T., Kabir, M. H., & Chae, O. (2010). Robust facial expression recognition based on local directional pattern. *ETRI journal*, *32*(5), 784-794.

Jeni, László A., Lőrincz, A., Nagy, T., Palotai, Z., Sebők, J., Szabó, Z., & Takács, D. (2012). 3D Shape estimation in video sequences provides high precision

evaluation of facial expressions. *Image and Vision Computing, 30*(10), 785–795.

Kabir, H., Jabid, T., & Chae, O. (2012). Local directional pattern variance (ldpv): a robust feature descriptor for facial expression recognition. *The International Arab Journal of Information Technology, 9*(4), 382-391.

Kamachi, M., Lyons, M., & Gyoba, J. (1998). The japanese female facial expression (JAFFE) database. *Retrived on August 2012 from http://www.kasrl. org/jaffe.html*.

Kanade, T., Cohn, J. F., & Tian, Y. (2000). Comprehensive database for facial expression analysis. In *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition* (pp. 46-53). IEEE. Retrieved on June 2012 from http://ieeexplore.ieee.org/xpls/abs_ all.jsp?arnumber=840611

Kobayashi, H., Tange, K, & Hara, F. (1995). Real-time recognition of six basic facial expressions. In *Proceedings of the 4th IEEE International Workshop on Robot and Human Communication* (pp. 179-186). Seoul, SK: IEEE.

Kotsia, I., & Pitas, I. (2007). Facial expression recognition in image sequences using geometric deformation features and support vector machines. *IEEE Transactions on Image Processing, 16*(1), 172–187.

Kumar, A. C. (2009). Analysis of unsupervised dimensionality reduction techniques. *Computer Science and Information Systems/ComSIS, 6*(2), 217-227.

Lajevardi, S. M., & Lech, M. (2008). Facial expression recognition from image sequences using optimized feature selection. In *23rd International Conference on Image and Vision Computing, New Zealand* (pp. 1-6). IEEE. Retrieved on August 2012 from http://ieeexplore.ieee.org/xpls/ abs_all.jsp?Arnumber=4762113

Liu, W. F., Li, S. J., & Wang, Y. J. (2009). Automatic facial expression recognition based on local binary patterns of local areas. In *WASE International Conference on Information Engineering:* Vol. 1 (pp. 197-200). IEEE. doi:10.1109/ICIE.2009.36

Liu, W. F., Wang, Y., & Li, S. (2011). LBP feature extraction for facial expression

recognition. *Journal of Information & Computational Science, 8*(2), 412–421.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*(2), 91–110.

Lucey, P., Cohn, J. F., Kanade, T., Saragih, J., Ambadar, Z., & Matthews, I. (2010). The extended cohn-kanade dataset (CK+): A complete dataset for action unit and emotion-specified expression. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops* (pp. 94-101). Sanfrancisco, CA: IEEE.

Lyons, M. J., Budynek, J., & Akamatsu, S. (1999). Automatic classification of single facial images. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 21*(12), 1357-1362.

Ma, L., & Khorasani, K. (2004). Facial expression recognition using constructive feedforward neural networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, *34*(3), 1588–1595.

Matsuno, K., Lee, C. W., Kimura, S., & Tsuji, S. (1995). Automatic recognition of human facial expressions. In *Proceedings of the Fifth International Conference on Computer Vision* (pp. 352-359). Cambridge, MA: IEEE.

Mehrabian, A. (1968). Communication without words. *Psychology Today*, *2*(4), 53-55.

Michel, P., & El Kaliouby, R. (2003). Real time facial expression recognition in video using support vector machines. In *Proceedings of the 5th international conference on Multimodal interfaces* (pp. 258-264). New York: ACM.

Naika C.L., S., Jha, S., Das, P., & Nair, S. (2012). Automatic facial expression recognition using extended AR-LBP. In K. R. Venugopal & L. M. Patnaik (Eds.). *Wireless Networks and Computational Intelligence SE - 29* : Vol. 292 (pp. 244–252). Berlin: Springer.

Ojala, T., & Pietikäinen, M. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, *24*(7), 971–987.

Ojala, T., Pietikäinen, M., & Harwood, D. (1996). A comparative study of texture measures with classification based on featured distributions. *Pattern*

*Recognition, 29*(l). Retrieved on Sepetember 2012 from http://www.sciencedirect.com/science/article/pii/0031320395000674

Ojansivu, V., & Heikkilä, J. (2008). Blur insensitive texture classification using local phase quantization. In *Image and Signal Processing* (pp. 236-243). Berlin: Springer.

Pantic, M., & Rothkrantz, L. (2000). Automatic analysis of facial expressions: The state of the art. *Analysis and Machine Intelligence, 22*(12), 1424–1445.

Reignier, P. (1995). Finding a face by blink detection. *ECVNet, Retrived on June 2012 from http://www-prima. imag. fr/ECVNet/IRS95/node13. html*.

Shinohara, Y., & Otsuf, N. (2004). Facial Expression Recognition Using Fisher Weight Maps. In *Proceedings of the Sixth IEEE International Conference on Automatic Face and Gesture Recognition* (pp. 499-504). Seoul, SK: IEEE.

Smith, J. R., & Chang, S. F. (1996). Tools and techniques for color image retrieval. In *Electronic Imaging: Science & Technology* (pp. 426-437). International Society for Optics and Photonics. Retrieved on July 2012 from http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=101519

Stricker, M. A., & Orengo, M. (1995). Similarity of color images. In *IS&T/SPIE's Symposium on Electronic Imaging: Science & Technology* (pp. 381-392). International Society for Optics and Photonics. Retrieved on July 2012 from http://dx.doi.org/10.1117/12.205308

Subramanian, K., Suresh, S., & Venkatesh Babu, R. (2012). Meta-cognitive neuro-fuzzy inference system for human emotion recognition. In *the International Joint Conference on Neural Networks* (pp. 1-7). IEEE.

Sun, X., Xu, H., Zhao, C., & Yang, J. (2008). Facial expression recognition based on histogram sequence of local gabor binary patterns. In *IEEE Conference on Cybernetics and Intelligent Systems* (pp. 158-163). Chengdu, China: IEEE.

Tamura, H., Mori, S., & Yamawaki, T. (1978). Textural features corresponding to visual perception. In *IEEE Transactions on Systems, Man and Cybernetics, 8*(6), 460-473.

Tian, Y.-I., Kanade, T., & Cohn, J. F. (2001). Recognizing action units for facial

expression analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(2), 97–115.

Tong, Y., Liao, W., & Ji, Q. (2007). Facial action unit recognition by exploiting their dynamic and semantic relationships. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(10), 1683–1699.

Wang, H., & Ahuja, N. (2003). Facial expression decomposition. In *Proceedings of thev Ninth IEEE International Conference on Computer Vision:* Vol. 2 (pp. 958–965). doi:10.1109/ICCV.2003.1238452

Wu, Y.-K., & Lai, S.-H. (2006). Facial expression recognition based on supervised LLE analysis of optical flow and ratio image. In *Proceedings of International Computer Symposium, Taipei, Taiwan*. Retrived on October 2012 from http://nthur.lib.nthu.edu.tw/dspace/handle/987654321/42273

Xu, Q., Zhang, P., Pei, W., Yang, L., & He, Z. (2006). A facial expression recognition approach based on confusion-crossed support vector machine tree. In *International Conference on Intelligent Information Hiding and Multimedia Signal Processing* (pp. 309-312). Pasadena, CA: IEEE.

Yang, S., & Bhanu, B. (2012). Understanding discrete facial expressions in video using an emotion avatar image. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, *42*(4), 980–992.

Yin, L., & Wei, X. (2006). Multi-scale primal feature based facial expression modeling and identification. In *7th International Conference on Automatic Face and Gesture Recognition* (pp. 603–608). Southampton, UK: IEEE.

Zeng, Z., Pantic, M., Roisman, G., & Huang, T. (2009). A survey of affect recognition methods: audio, visual, and spontaneous expressions. *IEEE transactions on pattern analysis and machine intelligence*, *31*(1), 39–58.

Zhang, Z., Lyons, M., Schuster, M., & Akamatsu, S. (1998). Comparison between geometry-based and gabor-wavelets-based facial expression recognition using multi-layer perceptron. In *Proceedings of the Third IEEE International Conference on Automatic Face and Gesture Recognition* (pp. 454-459). IEEE. Retrieved on November 2012 from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=670990

Zhao, G., & Pietikainen, M. (2007). Dynamic texture recognition using local binary patterns with an application to facial expressions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(6), 915-928.

Zhou, X. S., & Huang, T. S. (2003). Relevance feedback in image retrieval: A comprehensive review. *Multimedia Systems*, *8*(6), 536–544.

# APPENDICES

# Appendix A

## LIBSVM Parameters

-s svm_type : set type of SVM (default 0)

0 -- C-SVC

1 -- nu-SVC

2 -- one-class SVM

3 -- epsilon-SVR

4 -- nu-SVR

-t kernel_type : set type of kernel function (default 2)

0 -- linear: u'*v

1 -- polynomial: (gamma*u'*v + coef0)^degree

2 -- radial basis function: exp(-gamma*|u-v|^2)

3 -- sigmoid: tanh(gamma*u'*v + coef0)

-d degree: set degree in kernel function (default 3)

-g gamma: set gamma in kernel function (default 1/number of features)

-r coef0: set coef0 in kernel function (default 0)

-c cost: set the parameter C of C-SVC, epsilon-SVR, and nu-SVR (default 1)

-n nu: set the parameter nu of nu-SVC, one-class SVM, and nu-SVR (def. 0.5)

-p epsilon : set the epsilon in loss function of epsilon-SVR (default 0.1)

-m cachesize: set cache memory size in MB (default 100)

-e epsilon: set tolerance of termination criterion (default 0.001)

-h shrinking: whether to use the shrinking heuristics, 0 or 1 (default 1)

-b probability_estimates: whether to train a SVC or SVR model for probability estimates, 0 or 1 (default 0)

-wi weight: set the parameter C of class i to weight*C, for C-SVC (default 1)

The k in the -g option means the number of attributes in the input data.

(http://www.csie.ntu.edu.tw/~cjlin/libsvm/)

# BIOGRAPHY

| | |
|---|---|
| **NAME** | **Mohammad Shahidul Islam** |
| **ACADEMIC BACKGROUND** | **M.Sc.** (Mobile Computing & Communication), University of Greenwich, U.K., 2008<br><br>**M.Sc.** (Computer Science), American World University, 2005<br><br>**B.Tech**. (Computer Science & Technology), Indian Institute of Technology- Roorkee, India, 2002 |
| **PRESENT POSITION** | **Asst. Professor & Head**<br>Department of Computer Science**,** Faculty of Science & Engineering, Atish Dipankar University, Dhaka, Bangladesh. |
| **EXPERIENCES** | **Asst. Professor & Head** (Jan 2011-Mar. 2011) Department of Computer Science and Engineering, Faculty of Science & Engineering, Green University, Dhaka, Bangladesh.<br><br>**Senior Lecturer** Mar 2010 - Oct 2011 Department of Electronics & Telecom. Engineering, Daffodil International University, Dhaka, Bangladesh.<br><br>**Lecturer** Jan. 2008 - Feb. 2010 Department of Electronics and Telecom. Engineering, Daffodil International University, Dhaka, Bangladesh. |

**PUBLICATIONS**

**Mohammad Shahidul Islam** and Surapong Auwatanamongkol. (2013). Gradient Direction Pattern: A Gray-Scale Invariant Uniform Local Feature Representation for Facial Expression Recognition. *Journal of Applied Sciences, 13*(6), 837-845.
eISSN: 1812-5662
pISSN: 1812-5654

**Mohammad Shahidul Islam** and Surapong Auwatanamongkol. (2013). Facial Expression Recognition Using Local Arc Pattern. (Accepted). *Asian Journal of Information Technology*.
**e**ISSN: 1993-5994
pISSN: 1682-3915

**Mohammad Shahidul Islam** and Surapong Auwatanamongkol. (2013). A Novel Feature Extraction Technique for Facial Expression Recognition. *The International Journal of Computer Science Issues, 10*(1), 9-14.
eISSN: 1694-0784
pISSN: 1694-0814