University of Exeter

Department of Computer Science

Emotion Classification Using Combinations of Texture Descriptors

Huthaifa Ziad Abuhammad

September, 2018

Supervised by Professor Richard Everson & Dr Jacqueline Christmas

Submitted by Huthaifa Ziad Abuhammad to the University of Exeter as a thesis for the degree of Doctor of Philosophy in Computer Science

I certify that all material in this thesis which is not my own work has been identified and that no material has previously been submitted and approved for the award of a degree by this or any other University.

(signature) .....

Contents

List of Figures

List of Tables

Publications

The materials presented in chapters 3 and 4 have been published in:

Abuhammad, H., & Everson, R. (2018, June). Emotional Faces in the Wild: Feature Descriptors for Emotion Classification. In International Conference Image Analysis and Recognition (pp. 164-174). Springer, Cham.

Chapter 1

Introduction

Contents

1.1 Motivation

Computers have become an important part of our lives. They have moved from being just equipment for managing business and office tasks to being a real partner in social communication and interaction with humans. The prevalence in the past

two decades of small computers such as laptops, mobile phones and tablet devices has played a significant role in making the computers a permanent companion of our lives and our relationships with others.

Our emotions critically affect all aspects of our lives, from how we live, work, learn and play, to our decisions, big and small. Facial expression has a significant role in our communication with others. Today, mobile phones and computers are a significant way to communicate with others, so machines have to perform lots of human tasks, and need to have more human-like capabilities. Human emotional intelligence depends on our ability to recognise not only our own emotions but also those of other people; to this end, smart devices and advanced AI (artificial intelligence) systems should have the capacity to understand our emotions and interact with humans emotionally. Human-computer intelligent interaction (HCII) is a growing field that aims to achieve that. Human faces are the most important part of the human body to express feelings. People around the world use similar facial expressions to express emotions, such as happiness, sadness, anger, disgust, surprise and fear. Facial expressions enable people to understand each other; sometimes without even one single word, or in other words, facial expressions are a global language.

Most proposed automatic facial-expression recognition methods have been based on unnatural facial expressions, using databases that have been built from acted facial expressions. The creators of those databases have asked models or actors to express some facial expressions, and the problem here that the real emotions are different than those posed.

Nowadays, facial emotional recognition systems have been used in several appli-

Figure 1.1: Basic facial expression recognition system pipeline

cations (**?**) such as:

1. Detection and treatment of of depression and anxiety (**?**).
2. EmotiChat (**?**) .
3. Smart homes (**?**).
4. Affective/social robots (**?**).
5. Emotientv (**?**).
6. EmoVu

1.2 Facial expression recognition systems

Facial expression recognition systems follow the same general steps as illustrated in figure 1.1. The first is to find the Region Of Interest (ROI), which is the face. Un-

wanted areas may badly affect classification accuracy. Many facial detection methods have been proposed, such as the Viola-Jones algorithm (**??**).

The next step is extracting the features from a detected face. The purpose of image feature extraction is to describe an image efficiently by extracting the essential values and reduce the data without losing any significant details. There are many ways to do that, some depending on image texture itself, while other methods describe the image geometrically.

The last step is to classify the extracted features. In this step, machine learning plays the primary role in finding the differences between feature groups and making a decision as to what a group of values refers.

Some facial expression reconnection methods may contain further steps to improve accuracy: for example, applying some pre-process techniques to enhance the image before feature extraction, or reducing the length of the extracted features vector.

1.3 Challenges

Although many methods have been proposed for facial expression recognition, many challenges and difficulties are still facing this field, especially with natural expressions.

1. The very similarity of facial expressions means that many people find it difficult to identify the differences between certain expressions, for example, fear and surprise. We did a study, asking people to identify a range of facial expressions, and we found a significant variation in people's ability to determine facial expressions particularly these representing emotions such as anger and fear.

2. There is variation amongst human faces and their ways to express their emotions. Some people exaggerate their expressions, while others try to hide them. Moreover, there are many primary facial expressions and some secondary expressions, and this may change according to cultures and nations.

3. Most of the facial expression datasets contain unnatural facial expressions and have been built depending on acted facial expressions by some models or actors. These expressions are not indicative of the way people do in real life.

4. We have found that all the studied facial expression recognition methods are computationally expensive because they usually require large feature vector dimensionally. This makes difficulty for real-time applications and hinders the achievement of good results on different datasets.

1.4 Contributions

- We used the Labelled Faces in the wild dataset to provide a new facial expression dataset (eLWF) by involving people in the labelling process. The new dataset contains natural facial expressions rather than most of the current datasets which depend on actors mimicking facial expressions.

- We show that a combination of three image feature extraction methods Locally Binary Patterns(LBP) **?**, Histogram of oriented gradients (HOG) (**?**), and Dense speeded up robust features (D-SURF) (**??**) gives better image description than using only one of them and that this improves the classification rate with SVM and Random forests.

- We propose a new method to identify the most relevant image regions for emotion classification.

- We describe a new weighted voting algorithm, in which the weighted predictions of classifiers trained on pairs of classes are combined with the weights learned using an evolutionary algorithm. This method yields superior results, particularly for the hard-to-distinguish emotions.

- We used smoothing techniques to reduce the video classification errors (noise), by relying on a set of sequential video frames. We have employed a video database called DynEmo (**?**) developed by psychologists in our experiments, which contains videos of real facial emotions, and we labelled the videos based on frames rather than time.

1.5 Publications

The materials presented in chapters 3 and 4 have been published in:

Abuhammad, H., & Everson, R. (2018, June). Emotional Faces in the Wild: Feature Descriptors for Emotion Classification. In International Conference Image Analysis and Recognition (pp. 164-174). Springer, Cham.

(**?**)

1.6 Structure of the thesis

The rest of the thesis is structured as follows.

Chapter 2 We present a brief overview of some related features extraction methods, together with Random forest and support vector machine classifiers. Moreover, we present a brief overview of three optimisation methods: Nelder-Mead simplex direct search, Bayesian optimisation and the Covariance Matrix Adaptation Evolution Strategy. In particular, we discuss related methods of facial expression methods which depend on image-appearance techniques.

Chapter 3

In this chapter, we introduce the new "Emotional Faces in the Wild" dataset (eLFW) (**??**), a citizen-labelling of 1310 faces from the Labelled Faces in the Wild data.

Chapter 4

In this chapter, we present an automated new approach for facial expression recognition of seven emotions. Three types of texture features (LBP, HOG and D-SURF) from static images are combined, and the resulting features are classified using random forests. We achieve better than state-of-the-art accuracies using multiple texture feature descriptors. The use of random forests allows identification of the most important feature types and locations for emotion classification. Regions around the eyes, forehead, sides of the nose and mouth are found to be most significant.

Like people, machine classification of the eLFW and the Karolinska Directed Emotional Faces data obtained from actors, and poorest results are obtained in distinguishing the sad, angry and fearful emotions.

We describe a new weighted voting algorithm, in which the weighted predictions

of classifiers trained on pairs of classes are combined with the weights learned using an evolutionary algorithm. This method yields superior results, particularly for the hard-to-distinguish emotions.

Chapter 5 In this chapter, we apply the proposed method in the chapter to DynEmo video databases, and we describe how smoothing positively affects classifier scores by reducing errors.

Chapter 6 This chapter reviews a summary of the proposed algorithms and the associated results presented in this thesis and directions for future work.

Chapter 2

Background

Contents

2.1 Introduction

In this chapter, we present the background necessary to appreciate our proposed method based on image texture and r Random forest. We show a generic overview of various main concepts, with the more specific details of different techniques in relevant chapters.

Facial expression recognition field was first started being discussed at the end of the last century (**?**). Since then, many researchers have proposed methods to improve the ability of machines to recognise human facial expressions.

2.2 Review of main classification emotions methods

2.2.1 Active shape model And Active appearance model

An active appearance model (AAM) (**???**) is a computer vision algorithm which depends on statistically finding the values which fit with a grey image's texture values, and making a statistical link with an active shape model (**???**). ASM was first proposed by **?** based on models created from sets of training examples called Point Distribution Models (PDM), which represent objects as sets of labelled points called landmark points. Figure 2.1 illustrates a PDM as an example of face points landmarking. AAM was first proposed by **?** for face analysis. Since then the method has been commonly used in computer vision applications such as face matching, tracking faces, medical image analysis and emotions recognition (**??????**).

The training set are normally labelled manually, each shape is represented as shown in equation 2.3. The first step is computing the mean points for all the training set to find the mean shape $\bar{\boldsymbol{x}}$.

$$\boldsymbol{x} = (x_0, y_0, x_1, y_1, ..., x_k, y_k, ..., x_{n-1}, y_{n-1})^T \tag{2.1}$$

where $(x_k, y_k)$is the position point k

AAM's idea is to combine a model of face shape variation with a model of the appearance variations of a shape-normalised face. To create new shapes from the mean shape $\bar{\boldsymbol{x}}$, ASM variates new shapes and textures using Principal Component Analysis (PCA). PCA is applied to all training data to calculate the eigenvectors of the covariance matrix. To generate a new shape, the following equation is used:

$$\boldsymbol{x} \approx \bar{\boldsymbol{x}} + \boldsymbol{P}_s \boldsymbol{b}_s \tag{2.2}$$

where $\bar{\boldsymbol{x}}$ the mean shape, $\boldsymbol{P} = (\boldsymbol{P}_1 \mid \boldsymbol{P}_2 \mid ... \mid \boldsymbol{P}_t)$ contains t eigenvectors of the covariance matrix and $\boldsymbol{b}$ is a t dimensional vector given by

$$\boldsymbol{b} = \boldsymbol{P}^T(\boldsymbol{x} - \bar{\boldsymbol{x}}) \tag{2.3}$$

All image shapes are normalised to the mean shape; then each image is warped to its control points. The same pose (translation, scale and rotation) values are used in shape normalisation so that we can sample the grey level information $\boldsymbol{g}$ from a shape-normalised face patch. By applying PCA to this data we obtain this model:

$$\boldsymbol{g} \approx \bar{\boldsymbol{g}} + \boldsymbol{P}_g \boldsymbol{b}_g \tag{2.4}$$

A further PCA is applied to correlated shape and grey-level variations, to obtain:

$$\boldsymbol{x} \approx \bar{\boldsymbol{x}} + \boldsymbol{Q}_s \boldsymbol{c} \tag{2.5}$$

$$\boldsymbol{g} \approx \bar{\boldsymbol{g}} + \boldsymbol{Q}_g \boldsymbol{c} \tag{2.6}$$

where $\bar{\boldsymbol{x}}.$ is the mean shape, $\bar{\boldsymbol{g}}.$ the mean texture, Q_s, Q_g are matrices describing the modes of variation derived from the training set and c are the model parameters.

By varying the elements of $\boldsymbol{c}$ in equation 2.6 and 2.4, new shapes and images will be generated. So c_i is the variance of the i_{th} parameter given by λ_i. To generate similar shapes to the original training data, limits of $\pm 3\sqrt{\lambda_i}$ have been applied to the parameter b_i.

Now if we were given a new image g_s, and we want to find the shape points which fit the image we need to vary $\boldsymbol{c}$ to generate new images by a set of model parameters, c, we can generate a hypothesis for the shape, x, and texture, g_m, of a model instance, then finding the most similar generated image for the appearance model to g_s by computing the difference, $\delta g = g_s - g_m$. This is an optimisation problem to find the best $\boldsymbol{c}$ efficiently, which will generate shape landmarks which describe the face parts.

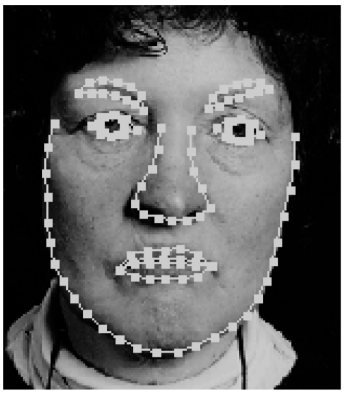


Figure 2.1: Point Distribution Model (**?**)

Since **?**, many researches have used AAM in many applications to recognise facial expression in f videos **??** and photos **???** . Facial expressions recognition is one the most discussed topics during the last three decades. Researchers have applied AAM to extract facial features face points to be classified using various classifiers. One of the ways that AAM has been used in facial recognition system is to detect action units as described in the Facial Action Coding System (FACS) **?**, which was based on minimal muscular movements and which individually or in combinations represents all facial expressions (**???**) By finding the action units existing in a face , and by using one or a group of them together, facial expression can be recognised. For an instance, AU1 is Inner Brow Raiser, AU2 is Outer Brow Raiser, AU15 Lip Corner Depressor and AU28 Lip Suck. To determine anger expression, for example, AU23 and AU24 must be present in the AU combination, where AU1+4+15 or 11 must be present for sadness **?**. To classify the extracted features to find the AUs trained classifiers have been used like neural networks trained with back propagation **?** on Cohn-Kanade AU-Coded (CK+) Facial Expression Database **?**, or Support Vector

Machine (SVM) in **?** with (CK+) database as well.

On the other hand, AAM was used to detect the facial expressions directly rather than AUs detection. The AAM features have been classified with several classifiers such as simple Euclidean-distance classification scheme **?**, K-means classification **?** or SVM **?**.

2.2.2 Image texture based methods

Image texture based or appearance-based methods have been widely used in computer vision applications face recognition and facial expressions recognition. The idea of image texture methods is using the image pixel values such as RGB, grey scale changes or illumination. Many methods have been proposed to describe the image and extract the image features such as Locally Binary Patterns(LBP), Histogram of oriented gradients (HOG) scale-invariant feature transform (SIFT) and speeded up robust features (SURF).

2.2.2.1 Local histogram of oriented gradients (HOG)

Local histogram of oriented gradients (HOG) is a method proposed by Dalal et al.(**?**). This method aims to describe an image with a set of local histograms. These histograms count occurrences of gradient orientation in a local part of the image. The HOG algorithm is similar to edge orientation histograms, scale-invariant feature transform descriptors, and shape contexts, but differs in that it is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalisation for improved accuracy(**?**). Figure 2.3 shows a HOG feature visualization for a face.

By focusing on the image, you can see the basic features of the image.

There are several primary steps for extracting HOG features: the first is Gamma/Colour Normalisation, where **?** found that gamma normalisation improves the classification rate. It is important to know that gamma normalisation gives a minor improvement so we can skip the step. The second step is computing the image gradients by applying the 1-D centred. Dalal and Triggs found that this gives the best results in one or both of the horizontal and vertical directions, $[-1, 0, 1]$ for vertical and $[-1, 0, 1]^T$ for horizontal. At every pixel we calculate a value for the x-derivative and another value for the y-derivative for x and y gradient magnitudes respectively, let us call them Sx and Sy. The equations defining the gradients are, respectively being:

$$S_x(i,j) = \frac{\partial I}{\partial x}(i,j) \tag{2.7}$$

$$S_y(i,j) = \frac{\partial I}{\partial y}(i,j) \tag{2.8}$$

where I is an image, and (i, j) the pixel coordinates.

The gradient magnitude itself M is computed as the square root of the quadratic sum of each gradient component, this is:

$$M(i,j) = \sqrt{S_x^2(i,j) + S_y^2(i,j)} \tag{2.9}$$

The gradient orientation angle is calculated by:

$$\theta(i,j) = arctan\left(\frac{S_x(i,j)}{S_y(i,j)}\right) \tag{2.10}$$

The third step is called Orientation Binning which aims to build a histogram of orientation for each cell; (where the image was subdivided into little cells). Each

pixel within the cell gives a weighted vote for an orientation-based histogram channel based on the values found in the gradient results. This histograms represent angles evenly spaced between 0° and 180° ("unsigned" gradient) or within 0° and −360° ("signed" gradient).

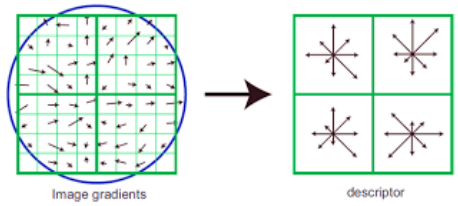


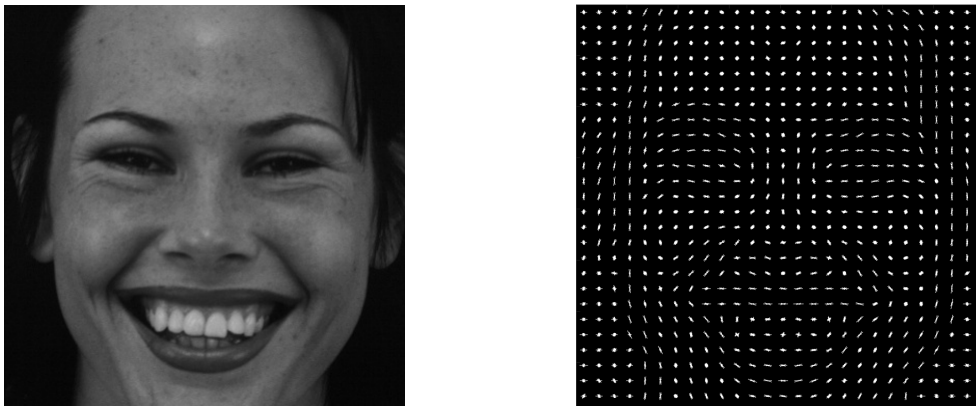
Figure 2.2: Image Gradients and Spatial/Orientation Binning.

The final step is block normalising histograms within each block of cells. Because of gradient strength variation as a result of local illumination variations and the foreground-background contrast, **?** found that some illumination normalization must be done to be essential for better accuracy. They explored different normalization schemes to achieve that. Let us define first v as the vector containing all the histograms for a given block, $\|v\|_k$ the k-norm of v with $k \in 1, 2$ and let ϵ be a small constant. The normalization schemes are:

$$L1 - norm : v \rightarrow \frac{v}{\|v\|_2 + \epsilon} \tag{2.11}$$

$$L1 - squarednorm : v \rightarrow \sqrt{\frac{v}{\|v\|_2 + \epsilon}} \tag{2.12}$$

$$L2 - norm : v \rightarrow \frac{v}{\sqrt{\|v\|_2^2 + \epsilon^2}} \tag{2.13}$$



(a) A woman's face (b) Visualization of extracted HOG

Figure 2.3: HOG descriptor Visualization

Dalal and Triggs' experiments found that L1-norm-squared and L2-norm performs similarly and achieves good results, but L1-norm decreases performance in a 5%. Not normalizing reduces enormously the performance by around 27%.

HOG method has been wildly used in computer vision to recognise objects. HOG features have used in facial expression recognition with multi-class RBF-SVM by extracting dense grid-based HOG features from images (**?**), where they used a cropped region from the aligned face and divided it into (48) squares 8 rows and 6 columns. In **?**, GEMEP-FERA dataset was used for training and testing 5 facial expressions anger, fear, joy, relief and sadness. The face has been divided into its main parts then extract each part's HOG features as shown in figure 2.4.

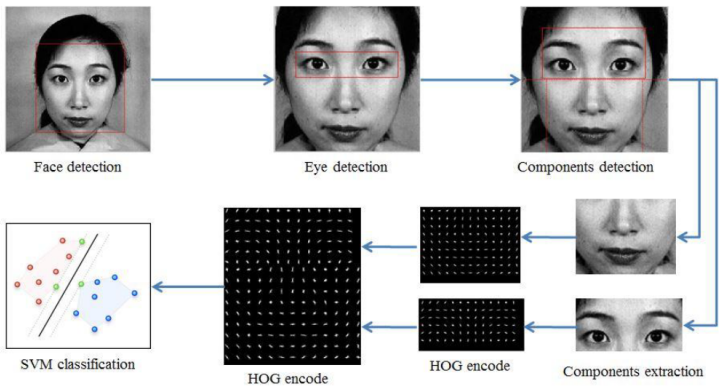


Figure 2.4: An overview of the face recognition system with HOG (**?**)

The HOG descriptor could be effectively exploited for facial expression recognition purposes, and the configuration of HOG parameters is able to give a strong image descriptor which allows a high classification performance for facial expressions (**?**).

2.2.2.2 Scale-invariant feature transform (SIFT) and speeded up robust features (SURF)

SIFT has been proposed by **?** the image rotation, affine transformations, intensity, and viewpoint change in matching features. The SIFT algorithm has 4 basic steps. First is to estimate a scale space extremum using the Difference of Gaussian (DoG). Secondly, a key point localisation must ne calculated where the key point candidates are localized and refined by eliminating the low contrast points. Thirdly, a key point orientation assignment based on local image gradient and lastly a descriptor generator to compute the local image descriptor for each key point based on image gradient magnitude and orientation **?**.

The main SIFT advantage is its stability for images in different resolutions, so it

gives good performance in machine vision applications. In facial expression methods, SIFT features represent the same object with different expression and illumination. Researchers have used SIFT descriptor with 2D and 3D images (**???**). **?** proposed a SIFT and SVM based method to investigate the robustness of SIFT features for various training images on face recognition and the used ORL and the Yale database for experiments and the found the method managed to handle the expression problems better than other algorithms at that time,figure 2.5 shows an example of images with extracted SIFT features.

Figure 2.5: An example of images with extracted SIFT features. The images represent the same object with different expression and illumination (**?**)

Speeded Up Robust Features (SURF) were first presented by Herbert Bay as novel scale- and rotation-invariant interest point detectors and descriptors (**??**). SURF is based on similar SIFT descriptor (**?**) properties. SURF is faster than SIFT and gives as good a performance as SIFT (**?**). (D-SURF) is a local feature detector and descriptor. It has been commonly used in computer vision tasks and object recognition classification. The D-SURF algorithm is based on the same principles and steps as SIFT, but the details in each step are different. The algorithm has three main parts: Firstly "interest point detection" was selected at important locations

in the image, for an instance, at T-junctions, corners and blobs. To achieve that, the algorithm uses a very basic Hessian matrix approximation because of its good performance in accuracy (**?**).The local next step is neighbourhood description, and matching is similar to the gradient information extracted by SIFT.

?

2.2.2.3 Local Binary Patterns (LBP)

Local binary patterns (LBP) method was first proposed by **?** as a texture descriptor depending on statistical analysis, and since then it has been widely used for face analysis due to their classification performance (**?**). The LPB operator compares each pixel in a 3x3 neighbourhood of the pixel to the central value and constructs a binary digit number from the result, thus computing local texture characteristics. One of the most important advantages of LBP features is their tolerance against illumination variation(**?**). Let us, therefore, define texture T in a local neighbourhood of a grayscale image as the joint distribution of the gray levels of $P+1(P>0)$ image pixels:

$$T = t(g_c, g_0, ..., g_{P-1}) \tag{2.14}$$

where g_c corresponds to the gray value of the centre pixel of a local neighbourhood. $g_p(p = 0, ..., P-1)$ is the gray values of P equally spaced pixels on a circle of radius $R(R>0)$ that form a circularly symmetric set of neighbours.

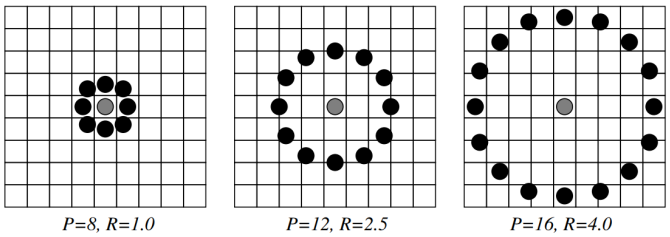


Figure 2.6: Circularly symmetrical neighbour sets. Samples that do not exactly match the pixel grid are obtained via interpolation.

Figure 2.6 shows three examples of different values of P and R. To achieve invariance with respect to any monotonic transformation of the grayscale, only the signs of the differences are considered:

$$T = t(s(g_0 - g_c), ..., (g_{P-1} - g_c)) \tag{2.15}$$

where corresponds to the grey value of the centre pixel (x_c, y_c), into the grey values of the 8 surrounding pixels, and function s(x) is defined as:

$$s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \tag{2.16}$$

The code characterizes the local image texture around (xc, yc):

$$LBP_{P,R}(x_c, y_c) = \sum_{p=0}^{P-1} s(g_p - g_c)2^n \tag{2.17}$$

The number of neighbours used to compute the basic LBP for each pixel in the input image is 8 neighbours. Figure 2.7 shows an example of LBP thresholding depending on 8 neighbours. The LBP operator is calculates the binary pattern

according to equation 2.16. The final decimal number calculated by the weights for each neighbour.

In LBP, the image is divided into a number of cells n_c which depends on the size of each cell. For example, if we have image I , and apply cell size c, the number of cells is calculated by the following equation:

$$n_c = floor(size(I)/c) \tag{2.18}$$

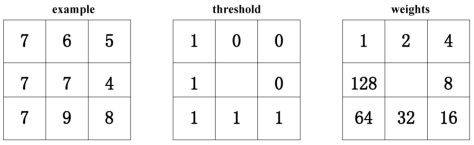


Figure 2.7: An example LBP thresholding depending on 8 neighbours.
Pattern: 11110001, Decimal = 1+16+32+64+128=241.

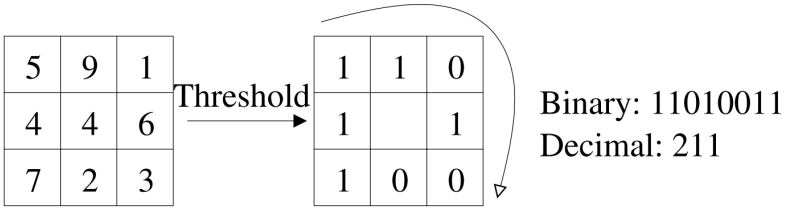


Figure 2.8: The LBP operator(?)

LBP has been used with many classifiers to recognise facial expression because of its advantages, i.e., its tolerance of monotonic illumination changes and its computational simplicity (?). ? used a simple Local Binary Patterns (LBP) with Support Vector Machine are (SVM). They have tested the extracted LBP features with linear,

polynomial and RBF kernels SVM to classify seven facial expressions. They used the Cohn Kanade Facial Expression Database which was produced by **?**, and contains faces of 100 university student from age 18 to 30 years. **?** compared their results with Gabor wavelets, and they have found that LBP with SVM gave better classification accuracy than Gabor wavelets, and saved computational resources. They also proved that LBP give good results with different resolutions, even low-resolution images(**?**). In the same area, **?** have done a comprehensive study for facial expression recognition methods based on Local Binary Patterns, and they found that LBP features are effective and efficient for facial expression recognition and it give good results with low-resolution images.

LBP has been used for frontal faces and for angle pose faces, (**?**) proposed a multi-view facial expression recognition method using some extensions including multi-scale local binary patterns (LBP^{ms}) and local gabor binary patterns ($LGBP$), and they have tested it on photos from multiple dataset (**?**) to see how see how head pose affects facial expression with SVM classifier.

During the last decade, researchers have prosed many methods to use LBP and LBP's extensions. Some of them have tried to divide the face into equal blocks into grids (**?**); others have tried to divide the face into its main parts: eyes, nose and mouth as in **?** who have proposed pyramidal local binary pattern (PLBP) operator to recognise six facial expressions. They have tested the extracted features with four classifiers: nearest neighbours (2NN), Random forest (RF), SVM and decision tree. They used in experiments two datasets: CK+ and FG-NET.

2.3 Classification

In this thesis we will use two popular data classification methods, random forests (RF) (**?**) and support vector machine (SVM) (**?**). An RF model uses the bootstrap method to build n_{tree} randomly, then they are provided with randomly selected samples of the training input. The trees will work together to give the final decision by combining them together in a forest. SVM aims to find a hyperplane (linear decision surface) which divides data into two classes and has the largest distance between the closest elements of the two classes to each other. These elements are called vector machines. It is not easy always to find a linear decision surface, so the radial basis function (RBF) is used as a kernel after the data has been mapped into a higher dimensional space (**?**).

2.3.1 Random forest

Random Forest (RF) is a popular method in computer vision because of its capacity to operate with large multi-class datasets and give high accuracy results (**?**). Because of its great generalization ability, it is very fast to train and parallelise (**??**). RF classifier contains a combination of tree classifiers, and each one is created by a random vector independently from the input vector. Each tree gives a unit vote for the most popular class to classify an input vector (**?**).

To understand Random the forest algorithm, we need to understand the basic idea of decision trees. Considering an input feature X and an output Y, we need to train a model to predict Y for given feature X. Decision trees are based on performing

predictions using a sequence of simple decisions by an ensemble of hierarchical binary decisions (**?**). In while the decision tree, the top node is called root connected with two children nodes, nodes at the bottom are called leaves, as shown in figure 2.9.

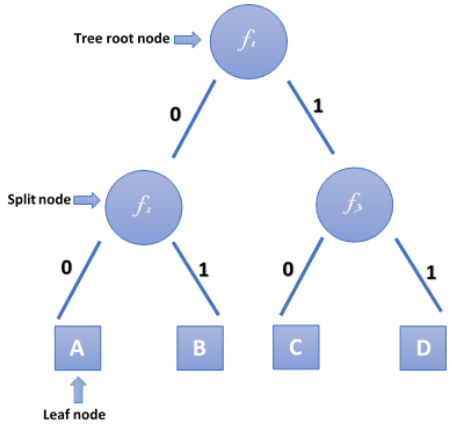


Figure 2.9: Simple decision tree

The main idea of Random forest (**?**) $\mathcal{F}$ is to make a group (ensemble) of T decision trees work together $\mathcal{F} = \{\mathbf{T_1}, ..., \mathbf{T_t}, ...\mathbf{T_T}\}$, where each tree node in random forests classifier is a weak classifier, each tree gets a "vote" in classifying (**???**). This combination of ensemble decorrelated trees provides very good generalization. So, from the same dataset, many random samples can be generated to be processed by constructing decorrelated trees. According to **?** randomization method "Bagging", for given dataset $\mathbf{D} = \{\mathbf{X}^{(n)}, Y^{(n)}\}_{n=1}^{N}$ is divided into random smaller dataset. Each data set S_t called bootstrap. By growing a tree $\mathbf{T_t}$ for each bootstrap, an ensemble decision trees working together to make vote for a new unseen observation.

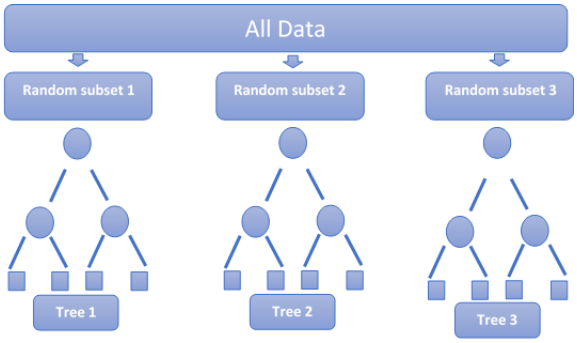


Figure 2.10: New subsets Randomisation by bagging method

In classification supervised training tasks, we have training data contains features and labels (output). Let given training dataset $\{(\mathbf{X^{(n)}}, Y^{(n)})\}_{n=1}^{N} \in \mathcal{X} \times \mathcal{Y}$, where $\mathbf{X} \subset \mathcal{X}$ and $\mathbf{Y} \subset \mathcal{Y}$, for $\mathcal{X} \subset \mathbb{R}^D$ the features space and the output space $\mathcal{Y} \subset \mathbb{R}$, which is a finite set of K discrete values $\mathcal{Y} = \{y_1, ..., y_k, ..., y_K\}$ represent the the classes . Let us consider the partition $\mathcal{P}_t = \{\mathcal{C}_t^{(zt)}\}_{z_t=1}^{Z_t}$ built by the random tree $\mathbf{T}_t$ (?). Class posteriors can be simply approximated in each cell $\mathcal{C}^{(zt)}$ of $\mathcal{P}_t$ as follows:

$$P(y_k \mid \mathbf{X} \in \mathcal{C}_t^{(z_t)}, \mathcal{P}_t) = \frac{\mid \{\mathbf{X}^{(n)} \in \mathcal{C}_t^{(}z_t), Y^{(n)} = y_k\} \mid}{\mid \{\mathbf{X}^{(n)} \in \mathcal{C}_t^{(z_t)}\} \mid} \tag{2.19}$$

In classification tasks, if a trained model is given a new unseen feature and it should predict to which class does it refer by sending the unseen feature through all trees of the forest and combining tree posteriors. class prediction for a new observation is the class that yields the largest weighted average of the class posterior probabilities computed using selected trees only. For each class $c \in \mathcal{C}$ and each tree $t = 1, ..., T$. Prediction for new observation x computes $\hat{P}(c \mid x)$ to estimate posterior probability of class c using t. $\mathcal{C}$ is the set of all distinct classes in the training data:

$$\hat{P}_{bag} = \frac{1}{\sum_{t=1}^{T} \alpha_t I(t \in S)} \sum_{t=1}^{T} \alpha_t \hat{y}_t I(t \in S) \tag{2.20}$$

where $\hat{y}_t$ is the prediction from tree t in the ensemble, S is the set of indices of selected trees that comprise the prediction and α_t is the weight of the tree.

$$\hat{Y}_{bag} = argmax_{c \in \mathcal{C}} \{P_{bag}(c \mid x)\} \tag{2.21}$$

.

The RFs classifier has proficient power to gauge the importance of each features variable (predictor) (**?**) by calculating how much a prediction error increases or decrease when (OOB) data for that variable is permuted while all others are passed on unaltered. The computations are carried out tree by tree as the random forest is built (**???**).

Breiman has proposed a method to evaluate the variable importance by measuring the Mean Decrease Accuracy of the forest when the values of X_m are randomly permuted in the out-of-bag samples. Suppose we have M input variables. After each tree is built, the values of the m_{th} variable in the out-of-bag examples are randomly permuted, and the out-of-bag data is run down the corresponding tree. The classification given for each x_n that is out of the bag is saved. This is repeated for $m = 1, 2, ..., M$. At the end of the run, the plurality of out-of-bag class votes for x_n with the m_th variable noised up is compared with the true class label of x_n to give a misclassification rate. The output is the percent increase in misclassification rate as compared to the out-of-bag rate (with all variables intact).(**??**).

In classification problems, (OOB) error returns the weighted misclassification

rate.

oobError predicts classes for all out-of-bag observations.

The weighted misclassification rate estimate depends on the value of 'Mode'. If you specify 'Mode','Individual', then oobError sets any in bag observations within a selected tree to the predicted, weighted, most popular class overall training responses. If there are multiple most popular classes, error considers the one listed first in the Class Names property of the TreeBagger model the most popular. Then, oobError computes the weighted misclassification rate for each selected tree.

If you specify 'Mode','Cumulative', then ooError returns a vector of cumulative, weighted misclassification rates, where et* is the cumulative, weighted misclassification rate for selected tree t. To compute et*, for each observation that is out of the bag for at least one tree through tree t, oobError finds the predicted, cumulative, weighted most popular class through tree t. oobError sets observations that are in the bag for all selected trees through tree t to the weighted, most popular class overall training responses. If there are multiple most popular classes, error considers the one listed first in the ClassNames property of the TreeBagger model the most popular. Then, oobError computes et*.

If you specify 'Mode','Ensemble', then, for each observation that is out of the bag for at least one tree, oobError computes the weighted, most popular class over all selected trees. oobError sets observations that are in bag for all selected trees through tree t to the predicted, weighted, most popular class overall training responses. If there are multiple most popular classes, error considers the one listed first in the Class Names property of the TreeBagger model the most popular. Then,

oobError computes the weighted misclassification rate , which is the same as the final, cumulative, weighted misclassification rate.

2.3.2 Support Vector Machine (SVM)

SVM is a binary classifier, and it was first proposed by Vladimir Vapnik in 1979 and first published in 1995. The basic idea of SVM is to find the hyperplane that linearly separates the n-dimensional data into two classes. Data do not always accept linear separating, so **?** suggested to use four essential kernel functions, Linear, Polynomial, Radial Basis Function (RBF)and sigmoid. SVM training aims to build a model from training data to be able to predict new values in testing data **?**. So if we have labelled training data $(x_i, y_i), ..., i = 1$ where $x_i \in R_n$ and $y \in \{1, -1\}$ the SVM seeks to find a solution of the optimisation problem:

$$\min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi \tag{2.22}$$

is subject to

$$y_i(w^T \phi(x_i) + b) \geq 1\xi_i, \xi_i \geq, i = 1, ..., l \tag{2.23}$$

SVM aims to find a linear separating hyperplane with the maximal margin in the training data space. Hsu et al. suggested to use four basic kernels for higher dimensional space: linear, polynomial, sigmoid and Radial Basis Function (RBF). In this thesis, the SVM classifier is optimised using an RBF kernel by Bayesian optimization method which is a powerful method to optimise functions (**?**).

$$K(x_i, x_j) = exp(-\gamma \parallel x_i - x_j \parallel^2), \gamma > 0 \tag{2.24}$$

2.4 Summary

In this chapter, we have reviewed some of the basic concepts regarding facial expression recognition and related topics such as features extraction and classification necessary to appreciate our proposed facial expression recognition method. We talked about the background of facial analysis, and we briefly introduced currents techniques which might be useful for our thesis.

Chapter 3

Emotional faces in the wild database

Contents

3.1 Introduction

Training and testing facial expression methods need databases containing human facial expressions. Researchers in the psychological field and machine learning have built various databases. The main disadvantage of most of the available datasets

is that they contain unnatural expressions because the people in the databases are actors expressing their expression as they have been asked to show. In this thesis, we are keen to work with more natural facial expressions, to investigate how the trained model can work with various people who expressed their emotions spontaneously.

Within the past two decades, significant effort has been made to build databases for use in facial expression recognition systems. These databases have been used for machine training and testing purposes. Some of the current databases that have mostly been used in the past two decades are the Karolinska Directed Emotional Faces (KDEF) **?**, the AR database (**?**), the Japanese Female Facial Expression Database (JAFFE) (**?**), Cohn-Kanade facial expression databaseCohn-Kanade (CK) (**?**) and the extended Cohn-Kanade dataset (ck+) (**?**), the MMI facial expression database (**?**), the FG-NET Facial Expressions and Emotion Database (**??**) and the PUT Face Database (**?**).

The commonly used method of constructing these databases is to ask some actors or models to show the required facial expressions. So most researchers have used relatively unnatural datasets because in real life, natural facial expressions are different from those made by actors. The expressions of the human face are varied and show some differences between cultures and even between one person to another (**?**). Just as human sometimes find it difficult to recognise some facial expressions, machines also face the same challenges. In this chapter, we introduce the "Emotional Labelled Faces in the Wild" dataset (eLFW), a citizen-labelling of 1310 faces from the Labelled Faces in the Wild data (**?**). To collect this data, we built a website and asked the citizen to label photos from the LFW dataset according to the emotional

expression displayed. This chapter presents the process of the new dataset collection and labelling and shows some summary statistics of the dataset.

3.2 Data collection

Our work began with building a new natural facial expression database using a current database called Labelled Faces in the Wild (LFW) (**??**). The LFW is a database of facial photographs designed for analysing the problem of unconstrained face recognition. The dataset contains 13,233 images of faces collected from the web and aligned using deep funnelling (**?**). Each face has been labelled with the name of the person pictured. 1680 of the people pictured have two or more distinct photos in the database. The only constraint on these faces is that they were discovered by the Viola-Jones face detector (**?**). Figure 3.1 shows some examples of LWF photos. Since LFW photos were not labelled by emotion, our first goal was to build a website to collect some data and information from people around the world to label the facial expression. This website aimed to build an extensive database of real faces together with the emotion they are expressing. Many facial photos were shown to the website visitors, and they were asked to choose the emotion that best matched the emotion being expressed by the face. They could label as many or as few as they wished.

To evaluate the citizen labelling we also used faces from a commonly use posed-dataset, the posed Karolinska Directed Emotional Faces(KDEF)(**?**). Figure 3.2 shows some samples from it. KDEF contains a set of 4900 pictures of human facial expressions of emotion. The dataset contains 70 individuals, each displaying seven different emotional expressions, each expression being photographed (twice)

Figure 3.1: Examples of citizen labelling.

Photo Name	Happiness	Sadness	Anger	Fear	Disgust	Surprise	Contempt	Frustration	Neutral	Undefined	Entopy
Aaron_Peirsol_0001.jpg	1	0	0	0	0	0	0	0	0	0	0
Colin_Powell_0233.jpg	0	0	0.25	0	0.25	0.25	0	0	0.25	0	2
David_Anderson_0001.jpg	0.25	0	0.25	0	0	0	0	0	0.50	0	1.5
Chloe_Sevigny_0001.jpg	0.5	0	0	0	0	0	0.5	0	0	0	1

Table 3.1: Voting data sample

from five different angles. In our experiments with KDEF, we used only frontal faces, which means 70 images for each facial expression, 490 images in total.

Website visitors were shown have seen different photos chosen randomly from the two databases. The probability that displayed faces came from LFW was 0.9, so on average 1 in 10 of the faces for labelling came from KDEF data set. This allowed us to evaluate the accuracy of the citizens' labelling method. For each face, people were asked to determine the emotion displayed from amongst the following: happiness, sadness, anger, fear, disgust, surprise, neutral and don't know.

Figure 3.2: Faces samples from KDEF database

Since the labels assigned differed between annotators and images were presented in a randomised order, images were retained in a pool of images to be labelled until they had been assigned labels by at least four times and the consensus emotion that is the modal classification was sufficiently unequivocal.

The new dataset involves useful data form each of the website visitors, and all visitors votes will be classified as the table 3.1 which shows four visitors have voted on four photos, and the entropy for each photo votes. It is clear that the best entropy value for the first photo which was zero. That means the four voters have voted the same, which called the minimum entropy value. The second one, the entropy value was two, that means each voter differ from other, this case called the maximum entropy value. The third photo in the table shows that the entropy value was 1.5,

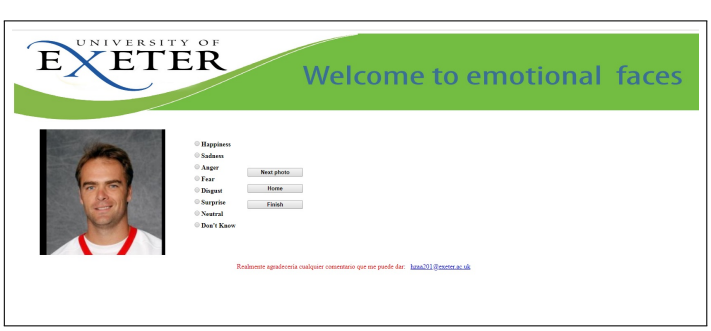
which means that two votes have voted one selection, and the other two have voted in another two selections. Finally, the last photo shows two selections have been voted by fifty per cent for each, and the entropy value is one.

Ambiguous classifications were avoided by calculating the entropy of the empirical distribution of classifications. Let p_n be the proportion, of citizens' votes for the n^{th} emotion class ($n = 1, \ldots, 8$), then the entropy, $H = -\sum_n p_n \log_2 p_n$, measures the agreement between the annotators. The entropy is maximised when all classes are assigned in equal proportion and is minimised when images are assigned to only a single class. We, therefore, kept an image in the pool of images to be labelled until the entropy of the citizens' assignments was less than 1 bit. Images that did not receive an unambiguous classification after 15 votes, and images for which the consensus was "don't know" were rejected.

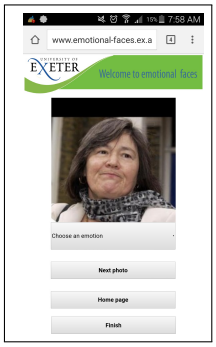
In addition to the LFW images, approximately one in ten images presented to the citizens was a KDEF posed image. This allowed us to check the integrity of individual annotators and, as discussed below, investigate human performance on the KDEF data. As Table 3.2 shows, the KDEF images were, unsurprisingly, easier than the LFW data for citizens to classify, requiring fewer votes to reduce the entropy below the acceptance threshold.

We used ASP.Net to build the website, and SQL Server to build the database. ASP.NET is a development framework for building web pages and websites with HTML, CSS, JavaScript and server scripting, while SQL Server is an efficient relational database management system (RDBMS) form Microsoft.

Two versions of the website have been designed to be more usable for all users,



(a) large screen version



(b) Mobile version

Figure 3.3: Screenshots two versions of the emotional faces website

those using PCs as shown in figure 3.3a, smartphones and tablet as shown in figure 3.3b. The website automatically redirects users to the appropriate version. This website has been tested many times, to make sure is it easy to use by users.

3.3 Summary of Citizens' classification

Table 3.2 shows summary statics of the new data set collection. 135 visitors correctly voted (1588) photos from both datasets, KDEF and LWF. To achieve low entropy values, the citizens made applicability seven votes on average on each LWF image and only five votes for KDEF. The average votes entropy for accepted LWF photos is 0.905, and for KDEF is 0.668. It is clear that, unsurprisingly, KDEF images were easier to classify than LWF images.

The citizens classified the 278 KDEF photos as shown in Table 3.3, which were distributed as: 49 fear, 35 anger, 42 disgust, 47 happy, 35 neutral, 34 sad and 36 surprised. The overall agreement of voters with the KDEF labelling was 80.6%.

Table 3.2: Some numbers from the built database

Users have voted	135
Accepted LWF photos	1310
Accepted KDEF photos	278
Mean number of votes for accepted KDEF photos	4.989
Average votes entropy for accepted KDEF photos	0.668
Mean number of votes for accepted LWF photos	7.203
Average votes entropy for accepted LWF photos	0.905

As the confusion matrix in Table 3.3 shows, there was complete agreement with the KDEF labelling for happy and neutral facial expressions, but only 42.9% for fear (confused principally with disgust and surprise), 77.8% for surprise (confused principally with fear and anger), and 78.6% for disgust (confused principally with sadness and surprise). The mean number of votes and average votes entropy for accepted KDEF photos values where larger than those for eLFW, which mean that it was easier for citizens to classify the KDEF images than the eLFW. Figure 3.4 shows examples of faces for which the citizen consensus differed from the KDEF labelling. We conclude that some facial expressions are similar to each other, and even humans may be confused while determining what the expression is, so machines may face the same problems while recognising the expression.

Table 3.3: Confusion matrices for the citizens' performance on KDEF images. True classes are shown by rows, with assigned classes in columns. FE: fear; AN: anger; DI: disgust; HA: happiness; NE: neutral; SA: sadness; SU: surprise.

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.429**	0.020	0.347	0.000	0.000	0.061	0.143
AN	0.057	**0.800**	0.086	0.000	0.000	0.029	0.029
DI	0.000	0.024	**0.786**	0.000	0.000	0.048	0.143
HA	0.000	0.000	0.000	**1.000**	0.000	0.000	0.000
NE	0.000	0.000	0.000	0.000	**1.000**	0.000	0.000
SA	0.059	0.000	0.059	0.000	0.000	**0.853**	0.029
SU	0.139	0.056	0.000	0.000	0.000	0.028	**0.778**

Figure 3.4: Examples of faces for which citizens' votes differ from the KDEF labelling. KDEF labels are shown above each image with the citizens' consensus below.

3.4 Summary

The lack of unposed labelled data has hampered work on machine recognition of emotional expressions. In this chapter, we have described the new emotional Labelled Faces in the Wild (eLFW) database, a citizen labelling of LFW faces. After labelling by citizens, the eLFW database comprises 190 fear images, 120 anger, 160 disgust, 330 happy, 240 neutral, 200 sad and 70 surprise images. The new data set enables us to evaluate the proposed texture based emotions classification on realistic data.

Chapter 4

Combining texture features for emotion classification

Contents

4.1 Introduction

Facial expression recognition is a rapidly growing research topic due to an increased interest in applications of human-computer interaction. As discussed in chapter 2, it has been studied extensively over the past decade, with much research concentrating on geometric features. Appearance based methods have become more prominent recently (**???**) and here we investigate the use of the combination of three feature descriptors, Histograms of Gradients (HOG) (**?**), Dense Speeded Up Robust Features (D-SURF) (**??**) and Local Binary Patterns (LBP) (**?**) for more accurate classification. We show that the combination gives a strong image descriptor. Classification with random forests, which embody natural feature selection, further allows us to find the location of the most important image descriptors.

In our proposed system, there are four main steps to extract and classify facial features: face detection, face alignment, facial texture feature extraction (LBP, HOG and D-SURF) and classification. We hypothesise that a combination of texture features is more effective than a single feature alone, yielding better classifications. In our experiments, we tested two state of the art classifiers, random forests and support vector machines (SVM). Our proposed system has two training phases: the first one uses random forests to locate the important facial regions by estimate feature

importance. The second training phase is to produce the final model by training with the important features only. The model automatically locates the important facial regions, which makes the classification faster and more accurate by excluding unnecessary and noisy face regions.

Image preparation and preprocessing steps are described in section 4.2. We describe the feature extraction and combination in section 4.3. Section 4.4 shows the baseline classification results for each one of the three feature extraction methods (HOG, LBP and D-SURF) with the two datasets (KDEF and eLFW), and then examines the effects of the combining texture features. Section 4.5 discusses the feature size reduction by determining only the important features and image masking. After that, we show random forest classification results applying the important features regions. Evidence of the similarity between machine and human classification is provided in section 4.7. Classification using SVMs is discussed in section 4.8.

The difficulties encountered by people and machines in distinguishing expressions displaying fear, anger and sadness leads us to consider alternative classifiers. So in section 4.9, we describe a pairwise random forest classifier in which the pairwise classifiers have a weighted vote to determine the overall class. We show how to optimise the weights using an evolutionary algorithm and present results showing the efficacy of the method. Finally, conclusions are drawn in section 4.10.

4.2 Image preprocessing

Image preprocessing is an important step to prepare images for feature extraction. Where the region-of-interest (ROI) is the face, we detect faces within the image to

Figure 4.1: Facial points detection

remove unwanted regions from images. Face alignment is then used to reduce the wide variety of face pose angles. Finally, all images are converted from RGB to grey scale.

- **Face detection and alignment**

  The first step is detecting the face. The face is detected using the Viola-Jones algorithm (**??**). The Viola-Jones algorithm can find faces, mouths and eyes. Having located the face, to align it we need to estimate the main facial points from the centres of the eyes and two points over the mouth as shown in figure 4.1. Bounding boxes around the eyes and mouth are created by the Viola-Jones algorithm. We then calculate the centre points of the eyes and mouth in a similar manner to (**?**) these equations:

$$(Cl_x, Cl_y) = (\frac{W}{4} + x, \frac{H}{2} + y) \tag{4.1}$$

$$(Cr_x, Cr_y) = (\frac{3W}{4} + x, \frac{H}{2} + y) \tag{4.2}$$

where Cl is the centre of the left eye, Cr is the centre of the right eye, W is the width of the bounding box, H is the height, and x and y are the pixel locations of the top-left corner of the the bounding box for the eyes.

All faces in the training dataset were aligned using the **?** method as shown in figure 4.3. This alignment method depends on a combination between deep learning into the congealing alignment framework to obtain features that can represent the image at differing resolutions based on network depth. It use a modified the learning algorithm for the restricted Boltzmann machine by combining a set sparsity penalty, which gives a topographic organization of the learned filters and improving subsequent alignment results **?**.

After alignment we estimated the face points, then we found the mean face shape by applying Procrustes analysis (**?**). The detected points for each image face were aligned to the mean shape by affine transformation (**?**), and each face was warped to its new aligned points.

Figures 4.2 and 4.3 show an example of a detected and face image after alignment and converting RGB values to grey then face detection.

- **Convert RGB image to greyscale.**

  Our proposed method works with greyscale images. LBP, HOG and D-SURF features were extracted from a greyscale image before classification. All input images in the datasets we used are colour images. Converting RGB values to

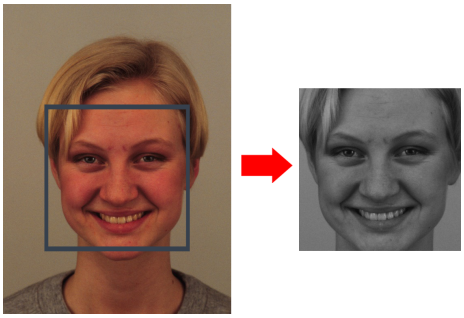


Figure 4.2: Face detection with Viola Jones

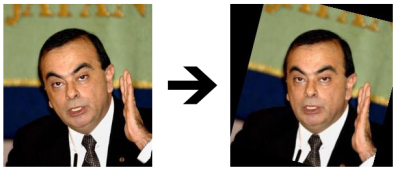


Figure 4.3: An example of facial alignment .

greyscale values by forming a weighted sum of the R, G, and B components: $0.2989 \times R + 0.5870 \times G + 0.1140 \times B$ (**?**).

4.3 Texture-feature extraction and combination

In machine learning and pattern recognition, feature extraction methods describe an image as a set of measured values called features. This data may be useful for further processing such as in machine learning. In this thesis, we applied three commonly used image descriptors, Local Binary Pattern (LBP), Histogram of oriented gradients (HOG) and Dense Speeded Up Robust Features (D-SURF), where we hypothesise that a combination of the three descriptors would give better image description than any single one. Local binary pattern (**??**) is a non-parametric descriptor which

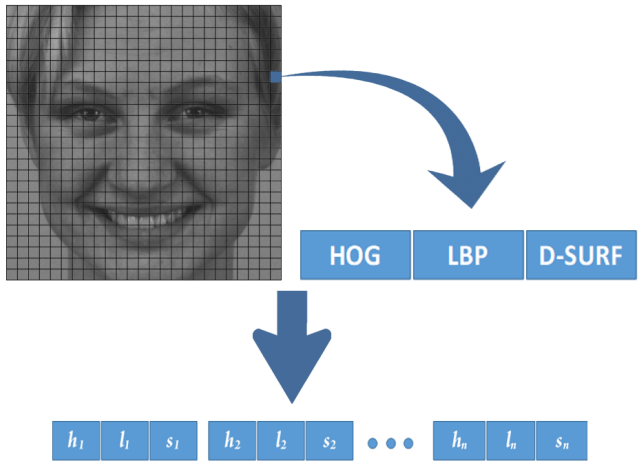


Figure 4.4: Block features (HOG, LBP and SURF) extraction and combination

efficiently summarises the local structures of images.

For an image $\mathbf{I}$ with size 400 by 400, we divide it into 25 by 25 non-overlapping blocks, which means 625 blocks. For each block the three feature descriptors: HOG $\mathbf{I_h}$, LBP $\mathbf{I_l}$ and D-SURF $\mathbf{I_s}$ are extracted and combined by concatenating each block's HOG, LBP and D-SURF descriptors in one vector $\mathbf{I_c}$.

$$\mathbf{I_c} = (\mathbf{I_{h_1}}, \mathbf{I_{l_1}}, \mathbf{I_{s_1}} \mathbf{I_{h_2}}, \mathbf{I_{l_2}}, \mathbf{I_{s_2}}, ..., \mathbf{I_{h_n}}, \mathbf{I_{l_n}}, \mathbf{I_{s_n}}) \tag{4.3}$$

. Figure 4.4 illustrates the division into blocks and feature extraction and combination.

The three features descriptors have been extracted for all the image blocks. The length of HOG for each block is 81, for LBP is 9 and for D-SURF is 64. So the length of the concatenated features is 81+64+9 = 154, and for the image $154 \times 625 = 96250$.

4.4 Baseline Random Forest Classification

4.4.1 KDEF experiments

Initially, we tested the performance of each one of the three features (HOG, LBP and D-SURF) separately with random forest classifier. 10-fold cross-validation was used ti classify the 490 images in the KDEF data. The results are shown in tables 4.1, 4.2 and 4.3. Table 4.1 shows the confusion matrix with only the HOG feature; the overall accuracy was 73%. Table 4.2 shows the confusion matrix result with the LBP features, where the overall accuracy was 79.9%. Finally, table 4.3 shows the D-SURF result with an overall accuracy of 70.3%.

The combined features were also tested with Random Forests and 10-fold cross-validation as well. Table 4.4 illustrates a visualisation of the performance. It is clear from the table that the combined model performance is emphatically better than using single one of the three features, achieving an overall accuracy of 89.8%.

Table 4.1: Confusion matrix for KDEF images with HOG featurs and a Random Forest classifier. True classes are shown by rows, with assigned classes in columns. The overall accuracy is 73%.

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.299**	0.042	0.028	0.042	0.113	0.142	0.328
AN	0.000	**0.771**	0.113	0.028	0.071	0.014	0.000
DI	0.000	0.042	**0.771**	0.085	0.0284	0.057	0.014
HA	0.0142	0.014	0.000	**0.914**	0.057	0.000	0.00
NE	0.000	0.099	0.000	0.000	**0.842**	0.028	0.028
SA	0.042	0.099	0.	0.014	0.227	**0.542**	0.028
SU	0.000	0.000	0.000	0.000	0.028	0.000	**0.971**

Table 4.2: Confusion matrix for KDEF images with LBP features and Random Forests classifier. The overall accuracy is: 79.9%.

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.456**	0.056	0.028	0.042	0.056	0.042	0.313
AN	0.000	**0.771**	0.113	0.028	0.071	0.000	0.014
DI	0.000	0.042	**0.914**	0.042	0.000	0.000	0.000
HA	0.000	0.000	0.014	**0.957**	0.028	0.000	0.000
NE	0.000	0.028	0.000	0.000	**0.942**	0.000	0.028
SA	0.028	0.028	0.042	0.085	0.142	**0.628**	0.042
SU	0.014	0.000	0.000	0.000	0.071	0.000	**0.928**

Table 4.3: Confusion matrix for KDEF photos with D-SURF features and Random Forests classifier. The overall accuracy is: 70.3%.

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.214**	0.085	0.057	0.057	0.100	0.171	0.314
AN	0.014	**0.714**	0.171	0.028	0.057	0.000	0.014
DI	0.000	0.043	**0.886**	0.057	0.000	0.014	0.000
HA	0.000	0.000	0.014	**0.957**	0.028	0.000	0.000
NE	0.014	0.071	0.000	0.000	**0.786**	0.000	0.128
SA	0.028	0.057	0.057	0.142	0.171	**0.442**	0.099
SU	0.028	0.000	0.000	0.000	0.043	0.000	**0.929**

Table 4.4: Confusion matrix for KDEF images with the combined features and Random Forests classifier. The overall accuracy is: 82.2%.

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.657**	0.129	0.043	0.000	0.014	0.100	0.057
AN	0.000	**0.829**	0.100	0.000	0.000	0.057	0.014
DI	0.000	0.057	**0.871**	0.000	0.000	0.043	0.029
HA	0.014	0.014	0.014	**0.943**	0.000	0.014	0.000
NE	0.000	0.000	0.014	0.071	**0.886**	0.014	0.014
SA	0.071	0.043	0.057	0.000	0.000	**0.800**	0.029
SU	0.157	0.029	0.029	0.000	0.000	0.014	**0.771**

4.4.2 eLFW experiments

To test how the three texture features work with real rather then posed emotions. As before we evaluated the performance using single feature types. 10-fold cross-validation was used with 5000 Random forest trees for the 1310 eLFW faces. The results are shown in tables 4.5, 4.6 and 4.7. Table 4.5 shows the confusion matrix eLFW database, and with only HOG feature, the overall accuracy was 73%. Table 4.6 shows the confusion matrix result with LBP feature, where the overall accuracy was 60.9%. Finally, table 4.7 shows the D-SURF result with an overall accuracy of 51.2%

It is clear from this section that with the LBP, HOG and D-SURF it was hard to classify the fear expression where happy and neutral are more easily classified in both datasets and fear is most often misclassified. The combination of the three texture features improved the overall accuracy with both datasets.

However, the overall accuracy for eLFW is lower than for KDEF, but the pattern of misclassification is the same, so it is clear the fear emotion for an instant was the lowest accuracy where the happy emotion was the higher for both dataset, KDEF and eLFW.

Table 4.5: Confusion matrix for eLFW images with HOG featurs and a Random Forest classifier. True classes are shown by rows, with assigned classes in columns. The overall accuracy is: 56.9%.

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.137**	0.158	0.195	0.058	0.084	0.184	0.184
AN	0.092	**0.608**	0.067	0.025	0.042	0.042	0.125
DI	0.069	0.044	**0.681**	0.063	0.044	0.050	0.050
HA	0.036	0.039	0.036	**0.742**	0.070	0.036	0.048
NE	0.042	0.054	0.071	0.083	**0.638**	0.046	0.054
SA	0.150	0.130	0.085	0.030	0.035	**0.465**	0.105
SU	0.086	0.043	0.029	0.057	0.043	0.071	**0.671**

Table 4.6: Confusion matrix for eLFW images with LBP features and Random Forests classifier. The overall accuracy is: 60.9%.

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.184**	0.153	0.195	0.047	0.074	0.179	0.168
AN	0.092	**0.650**	0.058	0.017	0.033	0.033	0.117
DI	0.069	0.044	**0.719**	0.050	0.038	0.038	0.044
HA	0.027	0.033	0.036	**0.773**	0.045	0.036	0.048
NE	0.025	0.033	0.050	0.079	**0.729**	0.042	0.042
SA	0.150	0.125	0.075	0.020	0.025	**0.500**	0.105
SU	0.114	0.071	0.057	0.057	0.043	0.071	**0.586**

Table 4.7: Confusion matrix for eLFW images with D-SURF features and Random Forests classifier. The overall accuracy is: 51.2%.

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.168**	0.153	0.195	0.058	0.079	0.179	0.168
AN	0.100	**0.508**	0.067	0.092	0.050	0.058	0.125
DI	0.081	0.056	**0.656**	0.075	0.050	0.038	0.044
HA	0.042	0.039	0.045	**0.645**	0.100	0.052	0.076
NE	0.042	0.054	0.058	0.079	**0.650**	0.063	0.054
SA	0.155	0.145	0.085	0.060	0.040	**0.375**	0.140
SU	0.114	0.086	0.057	0.114	0.100	0.114	**0.414**

Table 4.8: Confusion matrix for eLFW images with the combined features and Random Forests classifier. The overall accuracy is: 67.3%

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.232**	0.153	0.184	0.026	0.079	0.168	0.158
AN	0.092	**0.650**	0.058	0.025	0.017	0.042	0.117
DI	0.063	0.044	**0.738**	0.050	0.038	0.038	0.031
HA	0.012	0.006	0.021	**0.927**	0.024	0.003	0.006
NE	0.008	0.008	0.021	0.054	**0.879**	0.017	0.013
SA	0.150	0.135	0.075	0.025	0.020	**0.490**	0.105
SU	0.157	0.143	0.171	0.043	0.014	0.086	**0.386**

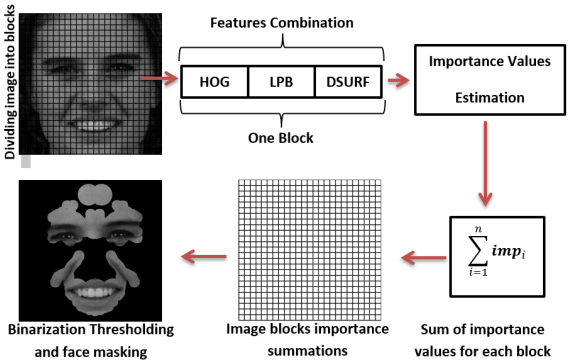


Figure 4.5: Diagram of image importance masking

4.5 Feature importance mask

After the three type of features have been extracted and combined, The features of images are trained by random forests to estimate the predictor importance for each value of the combined feature vector; see chapter 2. Then each image block is matched with its part of the combined vectors to decide where the important facial parts are. To locate the important face part, we sum the importance values for each face block, to yield a matrix of 25 by 25 combined importance vales. We applied the Otsu method (**?**) to covert the importance values to binarise. A mask of size 25 by 25 was produced. After getting the mask, by bicubic interpolation, we resized it to 400 by 400 to be the same as the size of training and testing images. Figures 4.5 and 4.6 illustrate the main steps of mask creation.

Figure 4.8 illustrates how a comparison between the random forest importance prediction values. Red colours show that HOG features were the most important in the location, while green colour refers to LBP and blue to D-SURF. It is clear that

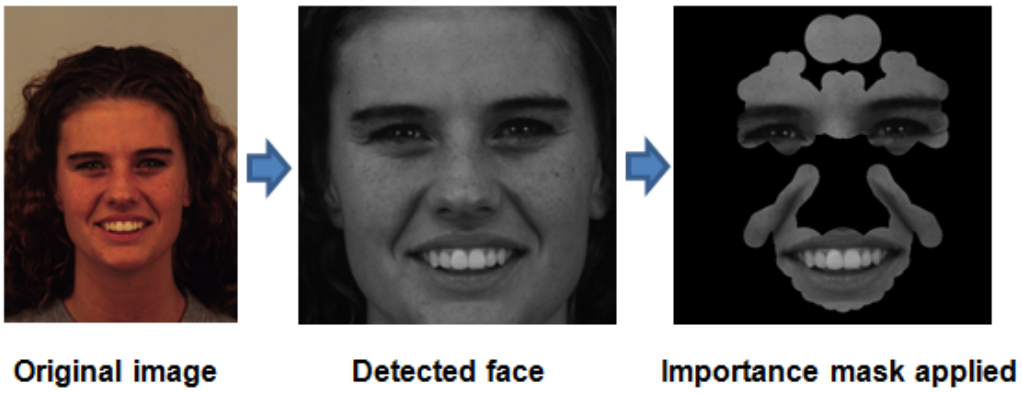


Figure 4.6: Importance masking steps for a KDEF image.

HOG and LBP occupy most the facial region compared to D-SURF. As Figure 4.7 shows, this procedure identifies the eyes, mouth, creases at the side of the nose and the forehead as most informative. We point out that this is an empirically determined mask rather than one chosen *a priori* and we obtain slightly different masks for the acted and wild faces.

Table 4.8 and 4.13 show two confusion matrices for that test, one with mask and one without. The accuracy on the eLFW data rose from 67.3% to 71.6% by using the importance mask. Fear and surprised expressions were the lowest accuracies. The model was still excellent with happy and neutral expression, and gave good results with anger and disgust, and acceptable with sad.

4.6 Random forest classification with importance mask

After obtaining the masks from the two datasets KDEF and eLFW, we applied each mask to all images in each dataset to remove the unwanted parts and keep only those

(a) eLFW mask (b) KDEF mask

Figure 4.7: Mask identifying the most important regions of the face for emotion classification derived by binarisng feature importance mas.

that were important. The resulting features were then classified using random forest (5000 trees). 10-fold cross validation was used to evaluate the classification scheme.

KDEF experiments

The table 4.9 shoes the confusion matrix for masked KDEF images with the combined features and Random Forests classifier. The overall accuracy is: 89.9%. The happy expression had the highest classification rate, while the fear expression had the lowest. With these two expressions, there is no significant improvement between masking and without masking.

Table 4.9: Confusion matrix for masked KDEF images with the combined features and Random Forests classifier. The overall accuracy is: 89.9%.

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.671**	0.143	0.043	0.000	0.000	0.143	0.000
AN	0.000	**0.857**	0.071	0.000	0.000	0.071	0.000
DI	0.000	0.043	**0.929**	0.000	0.000	0.029	0.000
HA	0.000	0.000	0.000	**0.986**	0.014	0.000	0.000
NE	0.000	0.000	0.000	0.014	**0.986**	0.000	0.000
SA	0.043	0.029	0.043	0.000	0.000	**0.886**	0.000
SU	0.014	0.000	0.014	0.000	0.000	0.000	**0.971**

Table 4.10: Comparison of classification accuracy of random forest classification using masked LBP, HOG and D-SURF texture features with other recent techniques. Evaluation on the KDEF data.

Method	Accuracy
?	74.05%
?	83.51%
?	84.07%
?	88.7%
Random forests with masked texture features	89.8%
Pairwise weighted random forests with masked texture features (§4.9)	95.1%

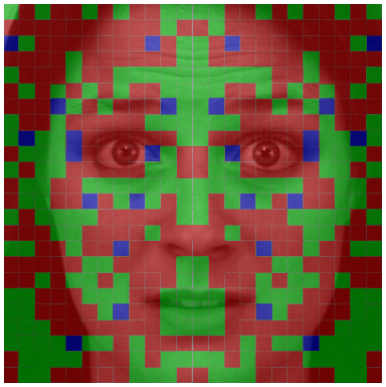


Figure 4.8: HOG, LBP and SURF importance values comparison.Red, blue and green indicate which feature type (HOG, LBP and D-SURF) was most important for each block.

The neutral and surprise rose significantly by applying the mask, 88.6% to 98.6% for neutral, and 77.1% to 97.1% for the surprise expression. Table 4.10 demonstrates that when evaluated on the KDEF data our proposed method using random forest classifiers and masked LBP, HOG and D-SURF features compares favourably with other state of the art techniques.

eLFW experiments

Importance masks are slightly different according to the training data images, so when we trained the system with KDEF photos, the mask was as shown in figure 4.6. The eLFW training produced a mask with some differences as shown in figure 4.7.

Table 4.13 shows a confusion matrix for masked eLFW images. It is clear that apply the mask significantly increased the accuracy from 67.3% without masking to

Table 4.11: Confusion matrix on eLFW database using proposed method, trained with KDEF. Overall accuracy was 74.7%.

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.371**	0.086	0.114	0.043	0.043	0.071	0.271
AN	0.057	**0.643**	0.157	0.000	0.014	0.057	0.071
DI	0.43	0.029	**0.743**	0.000	0.071	0.014	0.100
HA	0.029	0.014	0.000	**0.929**	0.014	0.014	0.000
NE	0.014	0.014	0.014	0.029	**0.900**	0.014	0.014
SA	0.029	0.014	0.043	0.000	0.029	**0.871**	0.014
SU	0.157	0.014	0.043	0.000	0.000	0.014	**0.771**

71.6% with masking (see table 4.8). The happy and neutral expressions are most easily classified in both datasets and fear is most often misclassified, particularly in the eLFW data, where it is confused with all the other classes except happy and neutral.

Table 4.10 shows a compassion between our proposed system with some recent works. It is seen by comparing our proposed system which gave the highest average accuracy 89.8% on average over other recent models. Some facial expressions are similar to each other, and even humans may be confused while determining what the expression is.

Table 4.11 shows a confusion matrix for testing the eLFW database after training our proposed system with the KDEF database. KDEF contains 70 faces for each expression. We chose 70 random faces from the eLFW for testing. It is clear from

Table 4.12: Average entropy of citizens' voting distributions for correctly and incorrectly classified eLFW images.

	Entropy (bits)		
Emotion	Correct		Misclassified
Fear	0.962	<	0.981
Anger	0.956	<	0.982
Disgust	0.983	=	0.983
Happy	0.777	<	0.902
Neutral	0.869	<	0.943
Sad	0.956	<	0.963
Surprised	0.937	<	0.976
Overall	0.920	<	0.961

the table that it gives good results in most facial expressions. Overall accuracy was 74.7%.

4.7 Comparison with citizen's classification

The patterns of misclassification for RF and and the citizen are similar. Table 4.12 shows the average entropy of the distributions of citizens' votes for images that were correctly and incorrectly classified. For each of the emotion classes, the average entropy for the misclassified images is greater than or equal to the average entropy of the correctly classified images displaying the same emotion. This indicates that

Table 4.13: Confusion matrix for random forest classification of the masked eLFW databases. The overall accuracy was 71.6%.

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.263**	0.153	0.200	0.016	0.042	0.168	0.158
AN	0.083	**0.725**	0.042	0.000	0.000	0.042	0.108
DI	0.063	0.031	**0.775**	0.038	0.031	0.031	0.031
HA	0.000	0.006	0.009	**0.973**	0.012	0.000	0.000
NE	0.004	0.004	0.017	0.054	**0.908**	0.008	0.004
SA	0.150	0.130	0.065	0.010	0.010	**0.535**	0.100
SU	0.157	0.129	0.157	0.029	0.000	0.086	**0.443**

there was more disagreement about the emotion displayed there was about correctly classified images.

4.8 Support vector machine performance

There are some parameters affect SVM with Radial Basis Function. The first and most relevant is the regularisation parameter C, which controls the trade-off between achieving a low error on the training data and minimising the norm of the weights. The second important parameter is cγ . To optimise the SVM, we need to find the best SVM parameters, Kernel coefficient γ and C. We used Bayesian optimization method which is a powerful method to optimise functions (**?**).

To achieve that we applied 10-fold cross-validation. The process starts by splitting

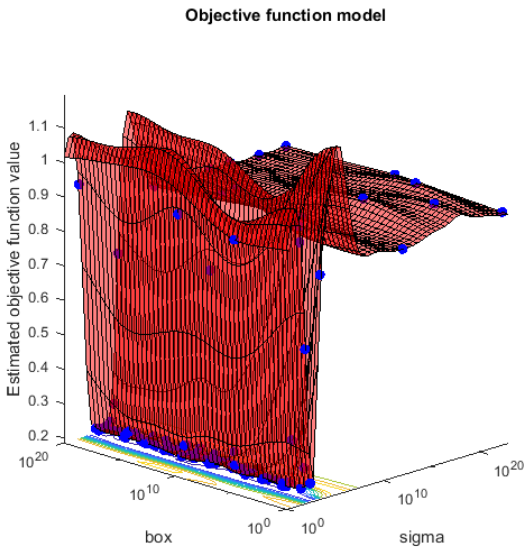


Figure 4.9: HOG cells, blocs and overlapping

the dataset into 10 smaller sets, to train the model using 9 of the folds as training data, then validates the model with the remaining part of the data. This approach can be computationally expensive but has a significant advantage by not wasting too much data as when fixing an arbitrary test set (**?**).

To optimise the SVM accuracy, we need to minimise the SVM loss. The best result was 0.28157, and γ was 4919.7 and C 0.11651.

Figure 4.9 shows how changing sigma and C parameters minimises the error. Classification loss functions measure the predictive inaccuracy of classification models.

Table 4.15 illustrates the same KDEF photos with Support Vector Machine classification (SVM) for whole facial features without applying the importance mask. To optimise the SVM, we need to find the best SVM parameters, Kernel coefficient γ and

Table 4.14: Confusion matrix for SVM classification of the eLFW database with the importance mask shown in Figure 4.6

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.237**	0.137	0.184	0.042	0.058	0.153	0.189
AN	0.058	**0.733**	0.025	0.000	0.008	0.075	0.100
DI	0.056	0.025	**0.750**	0.025	0.038	0.025	0.081
HA	0.018	0.021	0.15	**0.845**	0.033	0.018	0.048
NE	0.025	0.013	0.038	0.063	**0.829**	0.021	0.013
SA	0.175	0.125	0.060	0.010	0.025	**0.495**	0.110
SU	0.157	0.114	0.143	0.029	0.000	0.014	**0.543**

C. We used Bayesian optimization method which is a powerful method to optimise functions (**?**). The main idea of our optimisation is to minimise the cross-validation loss, by finding the best SVM parameters γ and C. The best parameters were where $\sigma = 6423.7$, and $C = 1.2333e + 19$. By comparing the outcome presented in table 4.15 with that shown in 4.4, the random forest gave slightly better outcomes. The SVM overall classification rate is 72%. After applying the KDEF mask, The SVM overall classification rate rose to 81% as shown in table 4.16. SVM was optimised and its parameters were $\sigma = 43482$, and $C = 9.2891e + 18$.

Table 4.14 illustrates a confusion matrix of testing the important features by SVM. Compared to the results in the right table 4.8, random forest still gives better results that SVM with the important combined features. According to table 4.14, the overall classification is 66.3% where the overall classification in the table 4.8 was

Table 4.15: Confusion matrix for SVM classification of the KDEF database without the importance mask. The overall accuracy is %72

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.643**	0.014	0.057	0.100	0.029	0.071	0.086
AN	0.043	**0.671**	0.071	0.000	0.043	0.086	0.086
DI	0.086	0.057	**0.714**	0.000	0.000	0.071	0.071
HA	0.000	0.000	0.000	**0.900**	0.057	0.014	0.029
NE	0.000	0.000	0.014	0.071	**0.843**	0.057	0.014
SA	0.068	0.043	0.071	0.000	0.029	**0.657**	0.114
SU	0.157	0.057	0.043	0.014	0.014	0.071	**0.643**

67.3%.

$$CVA = \sum_{i=1}^{K} \frac{A_i}{k} \tag{4.4}$$

where k = number of folds used, and Ai = accuracy measure of each fold.

Table 4.16: Confusion matrix for SVM classification of the KDEF database with the importance mask shown in figure 4.6. The ovrall accuracy is 81%.

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.714**	0.057	0.043	0.000	0.000	0.100	0.086
AN	0.043	**0.686**	0.071	0.000	0.029	0.086	0.086
DI	0.043	0.057	**0.786**	0.000	0.000	0.057	0.057
HA	0.000	0.000	0.000	**0.957**	0.029	0.000	0.014
NE	0.000	0.000	0.000	0.057	**0.900**	0.029	0.014
SA	0.057	0.057	0.043	0.000	0.000	**0.771**	0.071
SU	0.071	0.029	0.014	0.014	0.000	0.029	**0.843**

4.9 Weighted pairwise classification

Random forest classifiers combine decision trees which are naturally capable of multi-class classification, as opposed to dichotomous classifiers, such as SVMs, for which strategies such as one-versus-all or pairwise voting must be employed. In an effort to reduce the misclassifications of fear, anger, disgust and surprise, we also investigated the use of pairwise classifiers for classifying emotions. In this framework, a single classifier is trained to discriminate between a pair of classes. The item is then assigned to the class which receives the majority of votes from the pairwise classifiers.

For n-Class pairwise classification, the number of the classifiers is $c = \frac{n(n-1)}{2}$, so to classify the $n = 7$ emotions, there are 21 pairs of classifiers. Each pair of facial expressions class was used to train one random forest classifier. At the testing

Table 4.17: Posterior probabilities for two different fear images tested by the 21 random classifiers, the left has been wrongly classified and the right correctly classified.

Wrong classification decision

	Fear	Anger	Disgust	Happy	Neutral	Sad	Surprise	Total
Fear	x	0.87	0.88	0.85	0.49	0.62	0.38	**4.10**
Anger	0.13	x	0.80	0.73	0.12	0.22	0.12	**2.11**
Disgust	0.12	0.21	x	0.56	0.08	0.10	0.07	**1.13**
Happy	0.15	0.27	0.44	x	0.07	0.11	0.08	**1.12**
Neutral	0.51	0.88	0.92	0.93	x	0.64	0.48	**4.36**
Sad	0.38	0.78	0.90	0.89	0.36	x	0.31	**3.63**
Surprise	0.62	0.88	0.93	0.92	0.51	0.69	x	**4.55**

Correct classification decision

	Fear	Anger	Disgust	Happy	Neutral	Sad	Surprise	Total
Fear	x	0.94	0.93	0.86	0.85	0.69	0.48	**4.76**
Anger	0.06	x	0.78	0.77	0.27	0.16	0.10	**2.13**
Disgust	0.07	0.23	x	0.57	0.16	0.12	0.12	**1.27**
Happy	0.14	0.23	0.43	x	0.18	0.28	0.13	**1.40**
Neutral	0.15	0.73	0.84	0.81	x	0.19	0.31	**3.04**
Sad	0.31	0.84	0.88	0.72	0.81	x	0.31	**3.86**
Surprise	0.52	0.90	0.88	0.86	0.69	0.70	x	**4.54**

stage, each image was tested with all the 21 classifiers. For each facial expression, we calculate all votes from each classifier to get the total vote. The maximum vote for the seven expressions is the final decision.

As example, table 4.17 shows the posterior probabilities (see equation 2.21) of 21 random forest classifiers for two different images whose correct class was fear. Each cell in the tables is the posterior probability (classifier score) with the expression in the same row vs the expression in the same column. In the left table, the image has been wrongly classified as surprise (maximum total 4.55). Table 4.17 (right) shows another fear image which has been correctly classified (maximum total 4.76).

Table 4.18 shows the accuracy of RF and SVM classifiers on the KDEF data, using the usual 10-fold cross-validation. We noticed that both classifiers performance is very close to each other, and there is no significant difference between overall scores, while RF's average accuracy was 0.963 and SVM was 0.962. There is one advantage of SVM: all of the 21 classifiers gave an accuracy of over 90%, whereas RF gave two results lower than 90% with fear vs surprise, and neutral vs sad.

Table 4.20 shows the RF pairwise classification results of the KDEF data, by

Table 4.18: Pairwise RF classifiers and pairwise SVM classifiers performance with KDEF.

Classifier	**RF**	**SVM**	**Classifier**	**RF**	**SVM**
Fear vs Anger	0.956	0.935	Disgust vs Happy	0.976	0.978
Fear vs Disgust	0.968	0.978	Disgust vs Neutral	0.999	0.993
Fear vs Happy	0.976	0.978	Disgust vs Sad	0.960	0.942
Fear vs Neutral	0.940	0.957	Disgust vs Surprise	0.988	1.000
Fear vs Sad	0.920	0.900	Happy vs Neutral	0.999	0.993
Fear vs Surprise	0.883	0.900	Happy vs Sad	0.987	0.971
Anger vs Disgust	0.935	0.900	Happy vs Surprise	0.999	0.993
Anger vs Happy	0.993	0.985	Neutral vs Sad	0.895	0.940
Anger vs Neutral	0.952	0.950	Neutral vs Surprise	0.984	0.985
Anger vs Sad	0.936	0.950	Sad vs Surprise	0.976	0.978
Anger vs Surprise	0.994	1.000	**Overall**	**0.963**	**0.962**

comparing these results with the results in table 4.4, it is clear that pairwise classification has enhanced the overall accuracy from 89% to be 92%. We notice that the fear expression rate has significantly been improved. Table 4.10 demonstrates that when evaluated on the KDEF data our proposed method using random forest classifiers and masked LBP, HOG and D-SURF features compares favourably with another state of the art techniques.

In most pairwise architectures each of the constituent classifiers has an equal vote. Here we weigh the votes from each classifier and learn appropriate weights by optimising the classification accuracy of a validation set. More specifically, suppose

Table 4.19: Random Forestrs pair-classifiers optimised weights

	Fear	**Anger**	**Disgust**	**Happy**	**Neutral**	**Sad**	**Surprise**
Fear	x	0.198	0.202	0.159	0.106	0.159	0.176
Anger	0.226	x	0.247	0.232	0.450	0.127	0.122
Disgust	0.205	0.187	x	0.175	0.114	0.155	0.164
Happy	0.210	0.187	0.184	x	0.114	0.150	0.156
Neutral	0.216	0.187	0.174	0.107	x	0.157	0.159
Sad	0.206	0.195	0.172	0.106	0.158	x	0.164
Surprise	0.204	0.190	0.171	0.116	0.151	0.169	x

$y_{ij}(\mathbf{x}_n) \in [0, 1]$ is the output of the classifier discriminating between classes i and j for image features $\mathbf{x}_n$. Here we use random forests for each dichotomous classifier, so that $y_{ij}(\mathbf{x}_n)$ is the proportion of decision trees in the (i, j)-th forest that voted for class i. Then the overall score for class i is

$$Y_i(\mathbf{x}_n) = \sum_{j \neq i} \lambda_{ij} y_{ij}(\mathbf{x}_n) \tag{4.5}$$

where the weights are λ_{ij}, and the image is assigned to the class with the largest overall score: $\operatorname{argmax}_i Y_i(\mathbf{x}_n)$. The weights are constrained to be non-negative, $\lambda_{ij} \geq 0$ for all i and j, and we demand that $\sum_j \lambda_{ij} = 1$ for all i.

Training takes place in two phases. First, the constituent classifiers are independently trained on the pairs of classes. Secondly, the accuracy of the overall classifier on a second training data set is maximised by optimising the voting weights using an evolutionary optimiser. Here we used CMA-ES (**?**), a popular and effective evolutionary optimiser. Constraints were enforced by working in terms of variables θ_{ij}

Table 4.20: Confusion matrix for equally weighted pairwise classification on the KDEF data. Overall accuracy is 92%.

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.814**	0.029	0.043	0.000	0.000	0.029	0.086
AN	0.000	**0.886**	0.043	0.000	0.000	0.071	0.000
DI	0.000	0.043	**0.929**	0.000	0.000	0.029	0.000
HA	0.000	0.000	0.000	**0.986**	0.014	0.000	0.000
NE	0.000	0.000	0.000	0.014	**0.986**	0.000	0.000
Sad	0.029	0.029	0.043	0.000	0.000	**0.900**	0.000
SU	0.029	0.000	0.014	0.000	0.000	0.000	**0.957**

with

$$\lambda_{ij} = \frac{\theta_{ij}}{\sum_k \theta_{ik}}. \tag{4.6}$$

We note that the procedure is efficient because once the pairwise classifiers have been trained, classification scores $y_{ij}(\mathbf{x}_n)$ need only be calculated once before optimisation of the weights.

Table 4.19 shows the weights λ_{ij} obtained by the optimisation processes. Table 4.21 shows the confusion matrix obtained with the optimised pairwise classification using 10-fold cross-validation testing. The accuracies for the data sets have increased to 95.1% for KDEF and 76.6% for eLFW. As can be seen from the confusion matrix, classification accuracies of fear, anger, surprise and disgust have increased substantially, although for the eLFW data there is still considerable misclassification of fear (confused with disgust and surprise), sadness (confused with fear and anger) and

Table 4.21: Confusion matrices weighted pairwise classification of the KDEF and eLFW databases. The overall accuracy is 95.1% for KDEF and 76.6% for eLFW

KDEF

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.914**	0.029	0.000	0.000	0.000	0.000	0.057
AN	0.000	**0.943**	0.014	0.000	0.000	0.043	0.000
DI	0.000	0.014	**0.971**	0.000	0.000	0.014	0.000
HA	0.000	0.000	0.000	**0.986**	0.014	0.000	0.000
NE	0.000	0.000	0.000	0.014	**0.986**	0.000	0.000
SA	0.043	0.029	0.043	0.000	0.000	**0.886**	0.000
SU	0.014	0.000	0.014	0.000	0.000	0.000	**0.971**

eLFW

	FE	**AN**	**DI**	**HA**	**NE**	**SA**	**SU**
FE	**0.495**	0.089	0.142	0.011	0.042	0.074	0.147
AN	0.025	**0.792**	0.050	0.000	0.000	0.083	0.050
DI	0.056	0.019	**0.788**	0.044	0.031	0.025	0.038
HA	0.003	0.012	0.003	**0.970**	0.009	0.003	0.000
NE	0.004	0.013	0.000	0.017	**0.938**	0.029	0.000
SA	0.150	0.130	0.065	0.010	0.010	**0.535**	0.100
SU	0.086	0.086	0.071	0.014	0.129	0.086	**0.529**

surprise (confused with neutral, fear, anger and disgust). We remark that these emotions are all often expressed through a grimacing expression which may account for the difficulty in distinguishing them. Furthermore, these are the emotions about which the citizens showed the most disagreement (see section4.7).

4.10 Summary

Rather than using a single type of texture descriptor for appearance-based classification of emotions, we showed that a combination of LBP, HOG and D-SURF significantly increases classification accuracy. Furthermore, the random forest's feature selection was used to empirically identify the important regions of the faces for classifying emotion. As might be expected these are mainly around the eyes, mouth, creases on either side of the nose and the forehead. Use of our empirically defined importance mask enhances classification accuracy.

Further improvements to classification accuracy were obtained by pairwise weighted voting between dichotomous classifiers, and we showed how to learn optimal weights using an evolutionary algorithm. The resulting accuracies are significantly better than the current published state of the art results on the posed KDEF data. Nonetheless, particularly for unposed data, classification of fear, anger, disgust, sadness and surprise remains imperfect, and we obtain classification accuracies of about 77%. We observe that these are the emotions that humans find more difficult to classify from static images in the eLFW data.

Chapter 5

Facial expression recognition in Video

Contents

5.1 Introduction

In this chapter, we applied the proposed method in chapter 4 to the DynEmo video database. We investigated methods of smoothing the classifier predictions to exploit temporal continuity of emotions and therefore the classification error. Several smoothing techniques were investigated and optimised.

The field of video automatic facial expression analysis has overgrown in recent years. Nevertheless, most researchers still depend on databases that contain acted emotions from models or actors (**????**). It is clear by comparing the way facial expressions are expressed in real life are different than those were acted in most databases. A recent interesting database showing real facial expression is the DynEmo database (**?**) which was created by a multidisciplinary team of psychological researchers, computer scientists, statisticians, experimentation and instrumentation specialists, and a legal professional. The DynEmo database contains dynamic and natural emotional facial expressions, and filmed in natural but standardised conditions (**?**).

In this chapter, section 5.2 will show how the DynEmo database has been prepared to be usable in our work. In section 5.3 we show the result of applying the proposed method in the previous chapter to the new video database. Section 5.4 which shows how smoothing the classifier scores improves the accuracy. Finally, conclusions are drawn in section 5.5.

5.2 DynEmo database preparation

DynEmo database has been labelled over time not frame by frame. In our experiments, we need labelled frames to be used for training and testing. The researchers in the DynEmo database worked on the following facial expression expressions in the DynEmo database are Curiosity, happy, Surprise, Boredom, Disgust, Fright, Shame, Annoyance, Disappointment, Humiliation. To collect the data, the researchers have recorded films using hidden cameras while people (called encoders) were sitting on a chair at a small table facing the wall where a PC was projected on the wall using a video projector as illustrated in figure 5.1. During the video playback, experimenters were sitting in an adjacent room watching the encoders reaction. There recorded videos properties are width: 768, height: 576, frame rate: 25 frames per second, bits per pixel: 24 and video format:'RGB24'.

Table 5.1 shows a sample of original data layout in the DynEmo database. The most important columns for us in the data are C_Video which means the video name, E_Detectee is the labelled emotion, TC_Debut and TC_Fin the start and end time by milliseconds. Figure 5.2 shows an example for a labelled video on the time-line. On the DynEmo website all videos clips, but only 52 clips are available with labelling data. So in our work, we used the 44 clips as we mentioned. The first objective is to make the labelling based on frames rather than time. Moreover, to ensure that there is a consensus on the judgements on each frame. Work has started by preparing the database to be usable in training and testing.

Smiler to the work in chapter 3, we need to ensure that there is consensus on

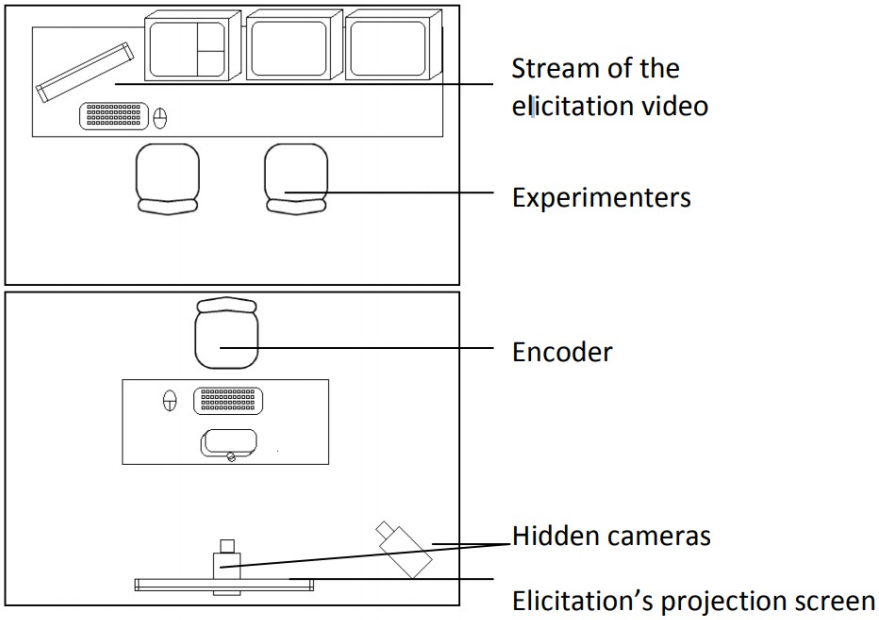


Figure 5.1: Design of the technical and recording rooms (**?**).

the facial expression on each frame. To achieve this, each frame must be assigned on emotion at least four different judges. To qualify the quality of the consensus we calculate the entropy value for the votes. The entropy must be less than 1 to accept the consensus vote. The entropy for n probabilities $(p_1, p_2, ..., p_n)$ is calculated by using this equation:

$$H = -\sum_{i=1}^{n} p_i \log_2(p_i) \tag{5.1}$$

where p; is the fraction of judges voting for class i.

We adapted the labelling way to be based in every single frame rather than the original method which was based on time sectors. Form the 52 videos available we got 44 videos that they include 14543 frames have been voted with consensus

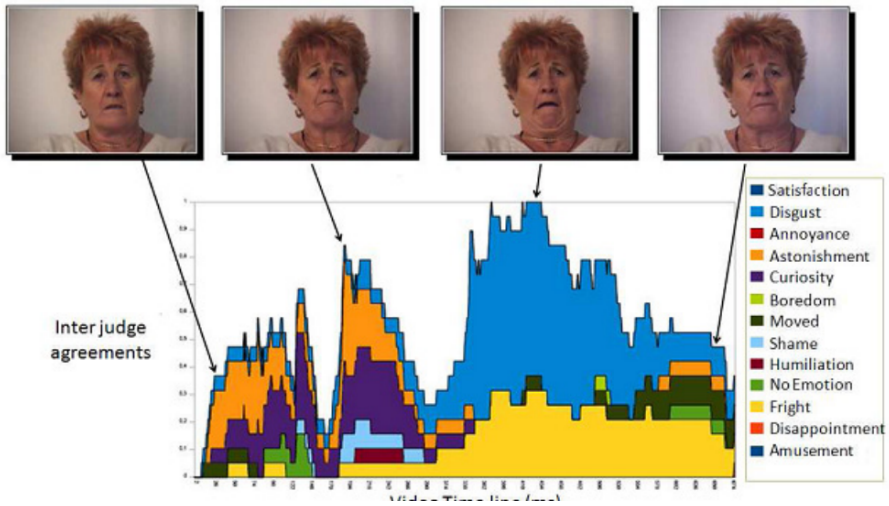


Figure 5.2: Emotional expressive time-line. Frames are taken from the video of an encoder who reported disgust and its corresponding underneath time-line (**?**).

($entropy < 1$). We removed some short parts of the 44 videos which have no consensus ($entropy > 1$). Some videos have multiple than one expression during the video, so we cut some short part from those videos which have not got consensus. Only five facial emotion remaining after calculating the entropy: Fear, anger, disgust, happy and surprise.

Natural facial expressions are vary even for the same person, for example, smile expression has many levels. The same person may express a small smile or wide smile. Some people try to hide their expressions sometimes, so this causes facial expression variation. This variation is a big challenge for natural facial expressions recognition. Figures 5.4a, 5.4b and 5.4c show happy expression variation for the same person from the DynEmo database. In most unnatural expression databases, the actors express a very similar way to show facial expression.

Table 5.1: DynEmo database data type.

C_INDUC	SEX_Sujet	SEXE_Juge	C_Juge	C_Video	E_Detectee	TC_Debut	TC_Fin
EM	F	N	237	DVD3_5.mpg	Stupefaction	7181	10183
EM	F	N	237	DVD3_5.mpg	Stupefaction	23438	29080
EM	F	N	237	DVD3_5.mpg	Stupefaction	39798	50305
EM	F	N	237	DVD3_5.mpg	Ennui	60543	64007
EM	F	N	237	DVD3_5.mpg	Stupefaction	96104	99176
EM	F	N	237	DVD3_5.mpg	Stupefaction	119316	123280
EM	F	N	133	DVD3_5.mpg	Stupefaction	4929	19463
EM	F	N	133	DVD3_5.mpg	Ennui	23241	30631
EM	F	N	133	DVD3_5.mpg	Ennui	66760	70727
EM	F	N	133	DVD3_5.mpg	Curiosite	87825	91947
EM	F	N	133	DVD3_5.mpg	Ennui	94415	95537
EM	F	N	133	DVD3_5.mpg	Curiosite	110200	117211
EM	F	N	133	DVD3_5.mpg	Curiosite	121754	123280
EM	F	N	229	DVD3_5.mpg	Stupefaction	6866	12845
EM	F	N	229	DVD3_5.mpg	Stupefaction	24385	30130

DynEmo preparation process has produced 44 videos have been labelled depending on the frame rather than time. Table 5.2 illustrates the new dataset, which contains 14543 frames distributed on five facial expressions as 8902 frames balled as happy, 313 as fear, 2192 as angry, 2665 as surprise and 471 as disgust. Figure 5.3 shows examples of new labelled frames. We used this obtained data in the experiments described in section 5.3.

Table 5.2: General statistics about the new database

Number of videos	Happy frames	Fear frames	Anger frames	Surprise frames	Disgust frames
44	8902	313	2192	2665	471

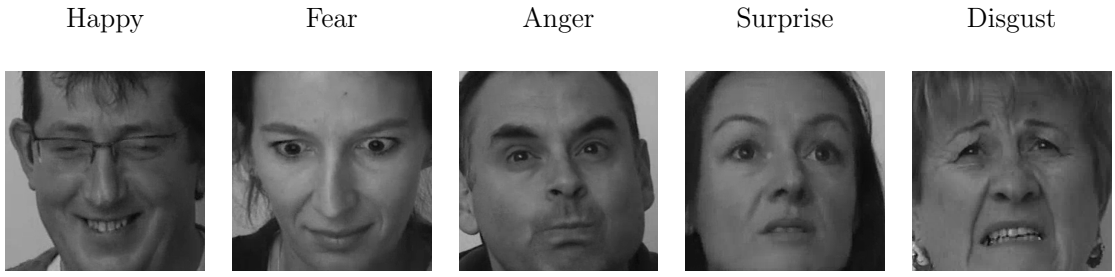


Figure 5.3: Examples of the five facial expression in the database

5.3 Video Classification Experiments

In our experiments, we used five facial expressions: happy, fear, angry, surprise and disgust. We trained our proposed system on videos and photos containing the five facial expressions. Because the videos are sequences of frames, we do not need to train the system using all frames, so we chose only one frame of each 25 from happy,

(a) Low smile. (b) Average smile. (c) Wide smile (laugh).

Figure 5.4: Happy expression variation for the same person.

anger and surprise and all fear and disgust frames, then we add to each class of the training data 70 images from KDEF and 70 images from eLFW. It is important to know that in the testing we have never used any video for testing and training at the same time. This training and testing were repeated five times (5-fold cross-validation), each time leaving different complete videos out for testing, and training with the remaining videos mixed with KDEF and eLFW images.

A trained Random forest model has been used to predict testing videos, and this returns scores for all training classes. Scores (posterior probability generated by each tree is a matrix with one row per observation and one column per class. Figure 5.5 shows a prediction behaviour of random frosts classier with two happy-labelled videos (DVD31_1 and DVD14_). Random Frosts returned voting values (scores) for each frame referring to the training classes. Table 5.3 right shows two confusion matrices, the left for one RF classifier and the right shows a confusion matrix for 10 pairwise classifiers. In the left table, the happy expression was the best rate with 85% accuracy, and fear like static images was the lowest accuracy with only 39%. The overall accuracy was 79.6%. The overall rise by pairwise classification in the right table to 83.4%, which is an increase of nearly 4%, similar to static images.

Like static images, our proposed system gives good results with videos as shown in table 5.3. In the next step in section 5.4 we investigate improving the performance of the classifiers by smoothing their scores, then we impose that the small misclassification should be fixed depending on the nearby frames.

Table 5.3: Confusion matrix on the DynEmo database using the proposed method, the left is by one RF classifier and the right 10-pairwise classifiers

Overall accuracy was 79.6%

	FE	**AN**	**DI**	**HA**	**SU**
FE	**0.390**	0.093	0.163	0.118	0.236
AN	0.045	**0.728**	0.175	0.018	0.034
DI	0.085	0.176	**0.609**	0.011	0.119
HA	0.044	0.043	0.022	**0.850**	0.040
SU	0.134	0.026	0.072	0.014	**0.753**

Overall accuracy was 83.4%

	FE	**AN**	**DI**	**HA**	**SU**
FE	**0.422**	0.093	0.150	0.118	0.217
AN	0.038	**0.776**	0.141	0.019	0.026
DI	0.064	0.174	**0.631**	0.011	0.121
HA	0.029	0.030	0.018	**0.890**	0.034
SU	0.114	0.024	0.068	0.014	**0.781**

5.4 Smoothing

It may be expected that the neighbouring frames in a video mostly contain the same facial expression. In other words, we assume that if the classifier has classified a frame as fear where the neighbouring frames were happy, it is likely to be a misclassification. To solve this problem, we suppose that smoothing the posterior probabilities of the classifier may reduce this misclassification and enhance the overall accuracy.

In smoothing, the individual data points (presumably because of misclassification) should be reduced, and the and points that are lower than the adjacent points are increased leading to a smoother data points(**?**). Smoothing reduces the classifier's misclassification according to a span of classifiers score values. If the classifier classified a majority of 100 sequential frames as happy, and 7 (sequential or not) of this 100 classified as sad, it is most likely that there is an error in classifying those seven, so we can rely on the nearby frames to solve this misclassification.

Figure 5.5: Classifier prediction behaviour on two happy-labelled videos

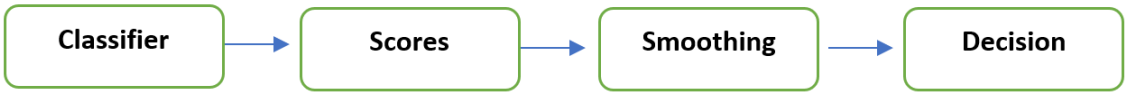


Figure 5.6: Decision making steps by smoothing the classifier scores).

accuracy.

5.4.1 Smoothing techniques overview

To smooth classifiers scores, several smoothing techniques have been tested: Locally weighted scatterplot smoothing (LOWESS and LOESS) and their robust versions, the Savitzky-Golay filter (**?**), and moving average smoothing.

Moving Average Filtering

The moving Average Filtering smoothed value is determined by neighbouring data points determined within a span. Moving Average Filtering smooth data by replacing each data point with the average of data points within the span. This method is

described by equation 5.2.

$$y_s(i) = \frac{1}{2N+1}(y(i+N)) + y(i+N-1) + ... + y(i+N-N) \tag{5.2}$$

where $y_s(i)$ is the smoothed value for the $i_th$data point, N is the number of neighbouring data points on either side of $ys_(i)$, and $2N+1$ is the span.

Savitzky-Golay filter (SGF)

The SGF is based on local least-squares polynomial approximation (**?**). The smoothed signal g(t) is calculated by convolving the signal f(t) with a smoothing (or convolution) function $h(t)$ for all observed data points p where $f(m)$ is the curve function at point m and $h(m-t) \neq 0(3)$. The convolution function $h(t)$ is defined for each combination of degree of the polynomial and window size (**?**).

Local Regression Smoothing

The names "lowess" and "loess" come from the term "locally weighted scatter plot smooth," as the two methods use locally weighted linear regression to smooth data. The method starts with computing the regression weights for each data point in the data window (span). In the next step, a weighted linear least-squares regression is performed. For lowess, the regression uses the first-degree polynomial. For loess, the regression uses a second-degree polynomial. Finally, The smoothed value is given by the weighted regression at the predictor value of interest. lowess and loess methods have robust versions include an additional calculation of robust weights, which is resistant to outliers values in the span.

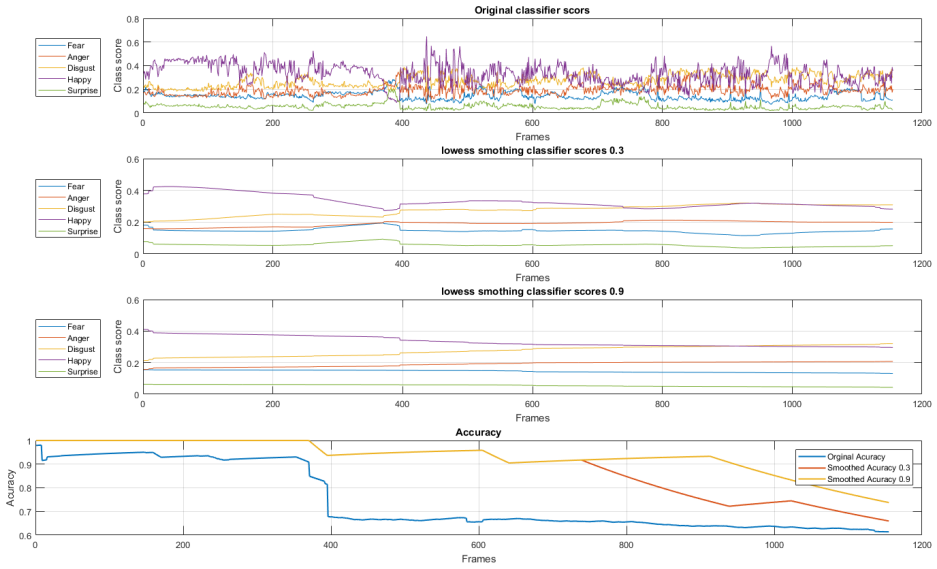


Figure 5.7: Lowess (linear fit) smoothing result for video DVD14_1_1.

5.4.2 Smoothing optimisation

Figure 5.8 shows an example of mooting a short video called (DVD14_ 1). The figure shows the original data in the top, followed by smoothing results with two arbitrary value span values. At the bottom of the figure, we can see the original and smoothed classification accuracy.

To smooth the classier scores, we need to find the optimal span value to get the best results. To achieve that, we use the Nelder–Mead method which is a popular numerical method to find the maximum or minimum of an objective function. However, the Nelder–Mead technique is a heuristic search method that can converge to non-stationary points on problems that can be solved by alternative methods (**?**).

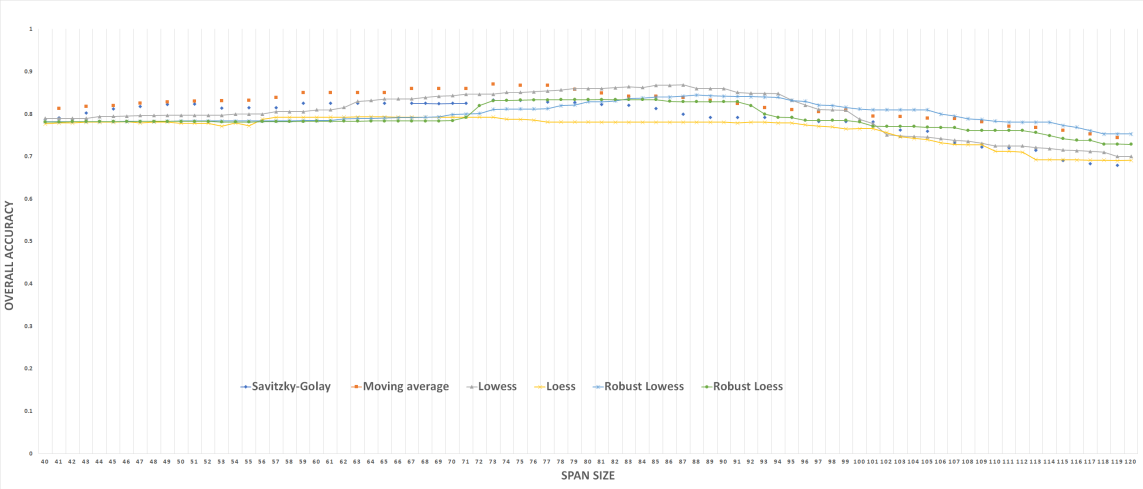


Figure 5.8: The overall accuracy vs. smoothing span_1_1.

By applying the Nelder–Mead on the classifier scores, we aim to find the optimal span which minimises the error. Table 5.4 shows the optimisation results for the all the smoothing methods we mentioned. The table shows the best window size (span) gives the best accuracy for each smoothing method. We found that the Moving average and Lowess gives the best smoothing accuracy. Moving average smoothing result was 87.9% where Lowess was 87%. Figure 5.8 illustrates the relationship between changing the smoothing span and the overall accuracy. We choosed a range between span size of 40 and 120 to show this figure, that means nearly 3 seconds and 8 seconds because the video rate is 15 frames per second. All the optimal span sizes are in this size range. It is clear that all smoothing methods give the best accuracy before around span size 90, then they begin to go down.

Table 5.5 shows two confusion matrices for the same the DynEmo database and by applying 5-fold cross-validation after applying the smoothing method. The right

Table 5.4: Optimising of smoothing span by Nelder–Mead (Staring point (0.5))

Smoothing method	**Optimal smoothing (span)**	**Overall accuracy**
Savitzky-Golay	69	0.8329
Moving average	**73**	**0.8709**
Lowess (linear fit)	87	0.8688
Loess (quadratic fit)	64	0.7935
Robust Lowess (linear fit)	88	0.8450
Robust Loess (quadratic fit)	81	0.8342

confusion matrix show on RF classifier and the left shows 10-pairwise classifiers. We can see that something improved both methods with marked increase, from 79.6% million (2015) to 87.1% for the one classifier, which is an increase of nearly 8%, and from 83.4% (2015) to 88.3% for the one classifier which is an increase of nearly 5%.

All five facial expression rate have been rose be smoothing.

Table 5.5: Confusion matrix on the DynEmo database using proposed method after smoothing, the left is by one RF classifier, and the right 10-pairwise classifiers

Overall accuracy was 83.4%

(Normal RF classifier)

	FE	**AN**	**DI**	**HA**	**SU**
FE	**0.457**	0.045	0.198	0.099	0.201
AN	0.043	**0.766**	0.145	0.012	0.034
DI	0.085	0.117	**0.679**	0.000	0.119
HA	0.014	0.009	0.018	**0.938**	0.021
SU	0.083	0.009	0.071	0.022	**0.815**

Overall accuracy was 88.3%

(10-pairwise classifiers).

	FE	**AN**	**DI**	**HA**	**SU**
FE	**0.543**	0.058	0.157	0.054	0.188
AN	0.031	**0.826**	0.124	0.010	0.010
DI	0.064	0.174	**0.641**	0.000	0.121
HA	0.013	0.011	0.009	**0.952**	0.016
SU	0.115	0.026	0.065	0.010	**0.784**

5.5 Summary

In this chapter, we used an existing psychological facial emotion video dataset called the DynEmo to prepare it to be used for machine training and testing purposes. We prepared 44 videos to include 14543 frames distributed on five facial expressions as 8902 frames balled as happy, 313 fear, 2192 angry, 2665 surprise and 471 disgust. We use the method which was proposed in chapter 4 to train random forests model with data from the DynEmo dataset mixed with some images from KDEF and eLFW. We tested some smoothing techniques to reduce the misclassification by smoothing the classifiers scores (posterior probability). To find the optimal smoothing span, we used the Nelder–Mead method to minimise the error.

As a result, like static images, our proposed system gives good results with videos. We found that applying smoothing methods with an optimal span value improved the performance of the classifiers by soothing their scores. As we have imposed, that the small misclassification should be fixed depending on the nearby frames. The best span size is between 60 and 90, that means 4 to 6 seconds. This effects on the ability of our proposed system to work in real-time applications because it needs 4 to 6 seconds to give the results. The best smoothing method is moving average which needs nearly 5 seconds to give the best result.

Chapter 6

Conclusion and perspectives

With the advancement in human-computer interaction, machines are becoming more important part of our lives. Facial expressions are an essential language to understand more about humans. One important factor that should be considered in developing a real facial expression recognition system is the availability of a useful database does not contain posed portraits of actors displaying emotions. Our work has started with the building of a website to construct a new natural facial expression database using a current database called Labelled Faces in the Wild (LFW) by asking citizens to vote what emotions they see in a selected group of images. Another current emotionally-labelled database has been used on the website called The Karolinska Directed Emotional Faces (KDEF) to evaluate citizens performance. The new database called Emotional- Labelled Faces in the Wild (eLFW).

In this thesis, we presented an automated new approach for facial expression recognition of seven emotions. Three types of texture features from static images are combined: Local Binary Patterns (LBP), Histogram of Oriented Gradients (HOG)

and Dense Speeded Up Robust Features (D-SURF), then the resulting features are classified using random forests. The use of random forests allows identification of the most important feature types and facial locations for emotion classification. Regions around the eyes, forehead, sides of the nose and mouth are found to be most significant. We classified the important features with random forest and support vector machines, and we found the classification performance became better than using all extracted facial features. We achieve better than state-of-the-art accuracies using multiple texture feature descriptors.

Further improvements to classification accuracy were obtained by pairwise weighted voting between dichotomous classifiers, and we showed how to learn optimal weights using an evolutionary algorithm. The resulting accuracies are significantly better than the current published state of the art results on the posed KDEF data. Nonetheless, particularly for unposed data, classification of fear, anger, disgust, sadness and surprise remains imperfect, and we obtain classification accuracies of about 77%. We observe that these are the emotions that humans find more difficult to classify from static images in the eLFW data.

We use the method which was proposed in chapter 4 to train random forests model with data from the DynEmo dataset mixed with some images from KDEF and eLFW. We tested some smoothing techniques to reduce the misclassification by smoothing the classifiers scores. To find the optimal smoothing span, we used Nelder–Mead method to minimise the error.

As a result, like static images, our proposed system gives good results with videos. We found that applying smoothing methods with an optimal span value improved

the performance of the classifiers by smoothing their scores. As we have imposed, that the small misclassification should be fixed depending on the nearby frames.

6.1 Future Work

More research effort is required to be put forth for recognising more complicated facial expressions, such as fatigue, pain, and mental states such as agreeing, disagreeing, lie, frustration, thinking as they have numerous application areas.

In this thesis, we have considered a real emotions recognition. For this mission we used existing data contains real emotion and we label it or modify the labelling method. It would be good in the future work if we create a new dataset for video and static images contain labelled data for face and body language.

Body language is very important to understand human emotions (**?**), and the tracking of the body language from video and static has received a large deal of attention over the years (**?**).

A more interesting avenue to explore would be to use convolutional neural networks for real emotion recognition which may improve the cost and time benefits and it may give more accurate results. Deep learning has an ability to generate new features from a limited series of features located in the training dataset. So we can work big data technology and save much time to work with more complex sets of features. More image texture features have to be explored to find if more accuracy may be archived by using them.

The real-time is a very important factor in any applications, as we saw in chapter 5, smoothing need nearly 5 seconds to get the result, so we need to investigate how

can we solve this problem in our proposed system. A possible solution may solve this issue is to explore Micro-Expression (**?**).

Bibliography

Abuhammad, H. and Everson, R. (2018). Emotional faces in the wild: Feature descriptors for emotion classification. In *International Conference Image Analysis and Recognition*, pages 164–174. Springer.

Aggarwal, J. K. and Cai, Q. (1999). Human motion analysis: A review. *Computer vision and image understanding*, 73(3):428–440.

Ahonen, T., Hadid, A., and Pietikäinen, M. (2004). Face recognition with local binary patterns. *Computer vision-eccv 2004*, pages 469–481.

Anderson, K. and McOwan, P. W. (2006). A real-time automated system for the recognition of human facial expressions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 36(1):96–105.

Barati, R. (2011). Parameter estimation of nonlinear muskingum models using nelder-mead simplex algorithm. *Journal of Hydrologic Engineering*, 16(11):946–954.

Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359.

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). Surf: Speeded up robust features. *Computer vision–ECCV 2006*, pages 404–417.

Belle, V. (2008). Detection and recognition of human faces using random forests for a mobile robot. *Master of Science Thesis, Academic Knowledge-based Systems Group*.

Berretti, S., Del Bimbo, A., Pala, P., Amor, B. B., and Daoudi, M. (2010). A set of selected sift features for 3d facial expression recognition. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 4125–4128. IEEE.

Borza, D., Danescu, R., Itu, R., and Darabant, A. (2017). High-speed video system for micro-expression detection and recognition. *Sensors*, 17(12):2913.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.

Breiman, L. (1999). Random forests—random features. *Technical Report 567*.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Breiman, L. (2002). Manual on setting up, using, and understanding random forests v3. 1. *Statistics Department University of California Berkeley, CA, USA*, 1.

Burgoon, J. K., Guerrero, L. K., and Floyd, K. (2016). *Nonverbal communication*. Routledge.

Candra, H. et al. (2017). *Emotion recognition using facial expression and electroencephalography features with support vector machine classifier*. PhD thesis.

Carcagnì, P., Del Coco, M., Leo, M., and Distante, C. (2015). Facial expression recognition and histograms of oriented gradients: a comprehensive study. *SpringerPlus*, 4(1):645.

Chen, J., Chen, Z., Chi, Z., and Fu, H. (2014). Facial expression recognition based on facial components detection and hog features. In *International Workshops on Electrical and Computer Engineering Subfields*, pages 884–888.

Chen, X., Udupa, J. K., Alavi, A., and Torigian, D. A. (2013). Gc-asm: Synergistic integration of graph-cut and active shape model strategies for medical image segmentation. *Computer Vision and Image Understanding*, 117(5):513–524.

Cohn, J. F., Zlochower, A. J., Lien, J. J., and Kanade, T. (1998). Feature-point tracking by optical flow discriminates subtle differences in facial expression. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference On*, pages 396–401. IEEE.

Cootes, T. F., Edwards, G. J., and Taylor, C. J. (1998). Active appearance models. In *ECCV98*, volume 2, pages 484–498.

Cootes, T. F., Edwards, G. J., and Taylor, C. J. (2001). Active appearance models. *IEEE Transactions on pattern analysis and machine intelligence*, 23(6):681–685.

Cootes, T. F., Edwards, G. J., Taylor, C. J., et al. (1999). Comparing active shape models with active appearance models. In *Bmvc*, volume 99, pages 173–182. Citeseer.

Cootes, T. F. and Taylor, C. J. (1992). Active shape models-'smart snakes'. In *BMVC*, volume 92, pages 266–275.

Cootes, T. F., Taylor, C. J., Cooper, D. H., and Graham, J. (1992). Training models of shape from sets of examples. In *BMVC92*, pages 9–18. Springer.

Cootes, T. F., Taylor, C. J., Cooper, D. H., and Graham, J. (1995). Active shape models-their training and application. *Computer vision and image understanding*, 61(1):38–59.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.

Dahmane, M. and Meunier, J. (2011). Emotion recognition using dynamic grid-based hog features. In *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 884–888. IEEE.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893. IEEE.

Davison, A. K., Yap, M. H., Costen, N., Tan, K., Lansley, C., and Leightley, D. (2014). Micro-facial movements: an investigation on spatio-temporal descriptors. In *European conference on computer vision*, pages 111–123. Springer.

Dhall, A., Goecke, R., Joshi, J., Hoey, J., and Gedeon, T. (2016). Emotiw 2016: Video and group-level emotion recognition challenges. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pages 427–432. ACM.

Dhall, A., Goecke, R., Joshi, J., Sikka, K., and Gedeon, T. (2014). Emotion recognition in the wild challenge 2014: Baseline, data and protocol. In *Proceedings of the 16th International Conference on Multimodal Interaction*, pages 461–466. ACM.

Dhall, A., Goecke, R., Joshi, J., Wagner, M., and Gedeon, T. (2013). Emotion recognition in the wild challenge 2013. In *Proceedings of the 15th ACM on International conference on multimodal interaction*, pages 509–516. ACM.

Dhall, A., Ramana Murthy, O., Goecke, R., Joshi, J., and Gedeon, T. (2015). Video and image based emotion recognition challenges in the wild: Emotiw 2015. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 423–426. ACM.

Edwards, G. J., Taylor, C. J., and Cootes, T. F. (1998). Interpreting face images using active appearance models. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, pages 300–305. IEEE.

Ekman, P. (1971). Universals and cultural differences in facial expressions of emotion. In *Nebraska symposium on motivation*. University of Nebraska Press.

Ekman, P. (1973). Cross-cultural studies of facial expression. *Darwin and facial expression: A century of research in review*, 169222:1.

Ekman, P. (1978). Facial expression. *Nonverbal behavior and communication*, pages 97–116.

Ekman, P. and Rosenberg, E. L. (1997). *What the face reveals: Basic and applied*

studies of spontaneous expression using the Facial Action Coding System (FACS). Oxford University Press, USA.

Essa, I. A. and Pentland, A. P. (1995). Facial expression recognition using a dynamic model and motion energy. In *Computer Vision, 1995. Proceedings., Fifth International Conference on*, pages 360–367. IEEE.

Fanelli, G., Gall, J., and Van Gool, L. (2011). Real time head pose estimation with random regression forests. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 617–624. IEEE.

Gross, R., Matthews, I., Cohn, J., Kanade, T., and Baker, S. (2010). Multi-pie. *Image and Vision Computing*, 28(5):807–813.

Hansen, N. (2006). The CMA evolution strategy: a comparing review. In Lozano, J., Larranaga, P., Inza, I., and Bengoetxea, E., editors, *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*, pages 75–102. Springer.

Hazewinkel, M. (2001). Affine transformation. *Encyclopedia of Mathematics, Springer*.

Hsu, C.-W., Chang, C.-C., Lin, C.-J., et al. (2003). A practical guide to support vector classification.

Huang, D., Shan, C., Ardabilian, M., Wang, Y., and Chen, L. (2011). Local binary patterns and its application to facial image analysis: a survey. *IEEE Transactions*

on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 41(6):765–781.

Huang, G., Mattar, M., Lee, H., and Learned-Miller, E. G. (2012). Learning to align from scratch. In *Advances in neural information processing systems*, pages 764–772.

Huang, G. B., Ramesh, M., Berg, T., and Learned-Miller, E. (2007a). Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst.

Huang, G. B., Ramesh, M., Berg, T., and Learned-Miller, E. (2007b). Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Technical Report 07-49, University of Massachusetts, Amherst.

Kanade, T., Cohn, J. F., and Tian, Y. (2000). Comprehensive database for facial expression analysis. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pages 46–53. IEEE.

Kanan, C. and Cottrell, G. W. (2012). Color-to-grayscale: does the method matter in image recognition? *PloS one*, 7(1):e29740.

Karami, E., Prasad, S., and Shehata, M. (2017). Image matching using sift, surf, brief and orb: Performance comparison for distorted images. *arXiv preprint arXiv:1710.02726*.

Kasinski, A., Florek, A., and Schmidt, A. (2008). The put face database. *Image Processing and Communications*, 13(3-4):59–64.

Kendall, D. G. (1989). A survey of the statistical theory of shape. *Statistical Science*, pages 87–99.

Khan, R. A. (2013). *Detection of emotions from video in non-controlled environment.* PhD thesis, Université Claude Bernard-Lyon I.

Khan, R. A., Meyer, A., Konik, H., and Bouakaz, S. (2013). Framework for reliable, real-time facial expression recognition for low resolution images. *Pattern Recognition Letters*, 34(10):1159–1168.

Ko, K.-E. and Sim, K.-B. (2010). Development of a facial emotion recognition method based on combining aam with dbn. In *Cyberworlds (CW), 2010 International Conference on*, pages 87–91. IEEE.

Kumari, J., Rajesh, R., and Kumar, A. (2016). Fusion of features for the effective facial expression recognition. In *Communication and Signal Processing (ICCSP), 2016 International Conference on*, pages 0457–0461. IEEE.

Learned-Miller, G. B. H. E. (2014). Labeled faces in the wild: Updates and new reporting procedures. Technical Report UM-CS-2014-003, University of Massachusetts, Amherst.

Liaw, A. and Wiener, M. (2002). Classification and regression by randomforest. *R news*, 2(3):18–22.

Louppe, G., Wehenkel, L., Sutera, A., and Geurts, P. (2013). Understanding variable importances in forests of randomized trees. In *Advances in neural information processing systems*, pages 431–439.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

Lozano-Monasor, E., López, M. T., Fernández-Caballero, A., and Vigo-Bustos, F. (2014). Facial expression recognition from webcam based on active shape models and support vector machines. In *International Workshop on Ambient Assisted Living*, pages 147–154. Springer.

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

Lundqvist, D., Flykt, A., and Öhman, A. (1998). The karolinska directed emotional faces (kdef). *CD ROM from Department of Clinical Neuroscience, Psychology section, Karolinska Institutet*, (1998).

Lyons, M., Akamatsu, S., Kamachi, M., and Gyoba, J. (1998). Coding facial ex-

pressions with gabor wavelets. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, pages 200–205. IEEE.

Martin, C., Werner, U., and Gross, H.-M. (2008). A real-time facial expression recognition system based on active appearance models using gray images and edge images. In *Automatic Face & Gesture Recognition, 2008. FG'08. 8th IEEE International Conference on*, pages 1–6. IEEE.

Martinez, A. M. (1998). The ar face database. *CVC Technical Report24*.

Martins, P., Sampaio, J., and Batista, J. (2008). Facial expression recognition using active appearance models. In *VISAPP (2)*, pages 123–129.

Mishra, S. and Dhole, A. (2015). A survey on facial expression recognition techniques. *International Journal of Science and Research*, 4(4):1247–1250.

Mockus, J. (2012). *Bayesian approach to global optimization: theory and applications*, volume 37. Springer Science & Business Media.

Moore, S. and Bowden, R. (2011). Local binary patterns for multi-view facial expression recognition. *Computer Vision and Image Understanding*, 115(4):541–558.

Ojala, T., Pietikäinen, M., and Harwood, D. (1996). A comparative study of texture measures with classification based on featured distributions. *Pattern recognition*, 29(1):51–59.

Ojala, T., Pietikainen, M., and Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 24(7):971–987.

Otsu, N. (1975). A threshold selection method from gray-level histograms. *Automatica*, 11(285-296):23–27.

Panchal, P., Panchal, S., and Shah, S. (2013). A comparison of sift and surf. *International Journal of Innovative Research in Computer and Communication Engineering*, 1(2):323–327.

Pantic, M., Pentland, A., Nijholt, A., and Huang, T. S. (2007). Human computing and machine understanding of human behavior: A survey. In *Artifical Intelligence for Human Computing*, pages 47–71. Springer.

Pantic, M., Valstar, M., Rademaker, R., and Maat, L. (2005). Web-based database for facial expression analysis. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pages 5–pp. IEEE.

Pauly, O. (2012). *Random Forests for Medical Applications*. PhD thesis, University of Munich.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2007). *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press.

Pu, X., Fan, K., Chen, X., Ji, L., and Zhou, Z. (2015). Facial expression recognition from image sequences using twofold random forest classifier. *Neurocomputing*, 168:1173–1180.

Rao, Q., Qu, X., Mao, Q., and Zhan, Y. (2015). Multi-pose facial expression recog-

nition based on surf boosting. In *Affective Computing and Intelligent Interaction (ACII), 2015 International Conference on*, pages 630–635. IEEE.

Ratliff, M. S. and Patterson, E. (2008). Emotion recognition using facial expressions with active appearance models. In *Proceedings of the Third IASTED International Conference on Human Computer Interaction,(Innsbruck, Austria)*, pages 138–143.

Santra, B. and Mukherjee, D. P. (2016a). Local dominant binary patterns for recognition of multi-view facial expressions. In *Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing*, page 25. ACM.

Santra, B. and Mukherjee, D. P. (2016b). Local saliency-inspired binary patterns for automatic recognition of multi-view facial expression. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 624–628. IEEE.

Savitzky, A. and Golay, M. J. (1964). Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8):1627–1639.

Scherer, K. R., Bänziger, T., and Roesch, E. (2010). *A Blueprint for Affective Computing: A sourcebook and manual*. Oxford University Press.

Setyati, E., Suprapto, Y. K., and Purnomo, M. H. (2012). Facial emotional expressions recognition based on active shape model and radial basis function network. In *Computational Intelligence for Measurement Systems and Applications (CIMSA), 2012 IEEE International Conference On*, pages 41–46. IEEE.

Shan, C., Gong, S., and McOwan, P. W. (2005). Robust facial expression recog-

nition using local binary patterns. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 2, pages II–370. IEEE.

Shan, C., Gong, S., and McOwan, P. W. (2009). Facial expression recognition based on local binary patterns: A comprehensive study. *Image and Vision Computing*, 27(6):803–816.

Simonoff, J. S. (2012). *Smoothing methods in statistics*. Springer Science & Business Media.

Soyel, H. and Demirel, H. (2012). Localized discriminative scale invariant feature transform based facial expression recognition. *Computers & Electrical Engineering*, 38(5):1299–1309.

Sung, J., Lee, S., and Kim, D. (2006). A real-time facial expression recognition using the staam. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 1, pages 275–278. IEEE.

Tcherkassof, A., Dupré, D., Meillon, B., Mandran, N., Dubois, M., and Adam, J.-M. (2013). Dynemo: A video database of natural facial expressions of emotions. *The International Journal of Multimedia & Its Applications*, 5(5):61–80.

Uijlings, J. R., Smeulders, A. W., and Scha, R. J. (2010). Real-time visual concept classification. *IEEE Transactions on Multimedia*, 12(7):665–681.

Van Kuilenburg, H., Wiering, M., and Den Uyl, M. (2005). A model based method for automatic facial expression recognition. In *Proceedings of the 16th European Conference on Machine Learning (ECML'05)*, pages 194–205. Springer.

Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE.

Viola, P. and Jones, M. J. (2004). Robust real-time face detection. *International journal of computer vision*, 57(2):137–154.

Wallhoff, F. (2006). Facial expressions and emotion database. *http://www. mmk. ei. tum. de/˜ waf/fgnet/feedtum. html*.

Whitehill, J., Bartlett, M. S., Movellan, J. R., and Emotient (2014). Automatic facial expression recognition.

Wu, Y., Wang, Z., and Ji, Q. (2013). Facial feature tracking under varying facial expressions and face poses based on restricted boltzmann machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3452–3459.

Yu, X., Huang, J., Zhang, S., Yan, W., and Metaxas, D. N. (2013). Pose-free facial landmark fitting via optimized part mixtures and cascaded deformable shape model. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1944–1951.

Yuqian, Z. and Bertram, E. (2016). Action unit selective feature maps in deep networks for facial expression recognition. In *The 2017 International Joint Conference on Neural Networks (IJCNN 2017)*. Ieee.

Zhang, L., Chen, J., Lu, Y., and Wang, P. (2008). Face recognition using scale invariant feature transform and support vector machine. In *Young Computer Scientists, 2008. ICYCS 2008. The 9th International Conference for*, pages 1766–1770. IEEE.

Zhou, S.-R., Yin, J.-P., and Zhang, J.-M. (2013). Local binary pattern (lbp) and local phase quantization (lbq) based on gabor filter for face representation. *Neurocomputing*, 116:260–264.