Universiteit Leiden

Computer Science

An Automated Pipeline for Face Recognition and Classification of Affect

Name:	N.B. Zuure
Date:	31/08/2017
1st supervisor:	Dr. F. W. Takes
2nd supervisor:	Prof. dr. A. Plaat

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

This thesis demonstrates a pipeline consisting of computer vision techniques and a neural network for automated support for classifying the still-face paradigm, a paradigm used by social scientists to follow development of interaction between parent and infant. The video recordings of this paradigm are taken at home and contain varying unstable conditions. These recordings are used for classifying positive and negative affect, sensitivity, intrusiveness and gaze. Classifying the aspects involved in the still-face paradigm is a time consuming task for social scientists since in a typical study there are hundreds of videos, and each one has to be watched from beginning to end. Here we focus on automated classification, where faces are extracted from videos and stabilized using a face detector and an active appearance model. A simple variant of the optical flow, a representation of motion between images, is calculated and then used as input for a neural network to automatically classify positive and negative affect for the infant and positive affect for the parent. These state of the art techniques are combined into a single pipeline to reduce minimal human interaction. The pipeline supports social scientists with their laborious manual classification. Our pipeline supports automated extraction, clustering and alignment of faces. These steps show promising results; noise in the data is reduced and stabilizing and rotating the faces causes them to be easily aligned. Further research is required for the remaining steps: calculating the optical flow and classification. Nevertheless, the initial steps pave the way for fully automated classification of affect and ultimately the still face paradigm.

Acknowledgements

I would like to thank my supervisor dr. F.W. Takes from LIACS, Leiden University. For his time, advice, ideas and the meetings we had. And I thank my second supervisor Prof. dr. A. Plaat, also from LIACS, for his ideas and feedback. Without their help this thesis would not be what it is.
Further I want to thank Prof. dr. J. Mesman from LUC, Leiden University. She provided the idea of this thesis as well as the required data. Besides her I want to thank N. Hendriks for his advice about about anonymizing images.
Special gratitude goes towards my parents for their support, faith and providing me the possibility to study.
Last but not least, I thank everybody that supported and or helped me throughout my study which I did not mention yet.

Contents

Chapter 1

Introduction

First in Section 1.1 an introduction of this thesis is given, then in Section 1.2 related work is discussed. Finally in Section 1.3 the structure of this thesis is given.

1.1 Introduction

The field of computer vision is growing larger and so is the number of applications using it. Where it started with detecting shapes and objects it has been used for much more complicated challenges such as the detection of brain tumours [17].

Detection of emotions and affect are an application of computer vision. Where it started with footage recorded under stable conditions and with posed emotions, it is now shifting towards “in the wild” situations [8]. These are no longer recordings from a lab with stable conditions, but are recordings taken outside. Emotions in the wild are remarkably more difficult to detect because of a few aspects: 1) the conditions are not stable which makes it harder to train models and 2) the emotions are mostly spontaneous and not posed, spontaneous emotions are harder to detect since facial changes are more subtle. Most of these detections methods have been trained and tested on adults and infants. However, models are trainable and can thus be used on infants as well.

Investigating infants is another field of research. This field is usually researched by social scientists who use paradigms to research the interaction and development of infants. These paradigms are recorded and their footage is investigated. One of these paradigms is the still-face paradigm (sfp), a paradigm using interaction between parent and infant. It consists of three phases: a baseline, a still-face and a reunion. The baseline exists of normal face to face interaction. During the still-face the parent maintains a neutral expressionless face, and is not allowed to respond to the infant. In the reunion phase the parent stops maintaining the expressionless neutral face and continues with the face to face interaction. This paradigm is (in our case) classified for the following aspects: sensitivity, intrusiveness, positive affect, negative affect and gaze. Classifying this paradigm is a time consuming task since every video needs to be recorded, watched and classified by a social scientist.

This thesis focuses on the classification of the sfp using state of the art methods from computer vision. It will be researched whether computers can classify both positive and negative affect for parent and infant. In the end, this is done to support social scientists with their classification and to reduce the time required of a social scientist for classification. The entire process is designed in such a way that human interaction is minimized. The things that have to be done manually consist of defining who is where in the video, and providing the start and end times of each phase.

1.2 Related work

In [23] different approaches for detecting Action Units (AU) on infants during the still-face paradigm are compared. Action Units are actions from facial groups of muscles i.e, raising of the inner brow. Groups of AU can be used to classify emotions. The used videos have been manually coded for each frame. The dimensionality reduction methods ”Principal Components Analysis with Large Margin Nearest Neighbour” (PCA+LMNN) and ”Laplacian Eigenmap” are compared, to see which one decreased the dimensionality of the features the best and which one improves the performance of classification. Also two classifiers are compared: Support Vector Machines (SVM) and k-nearest neighbour. The results show that PCA+LMNN with SVM lead to the best detection consistency.
Another study that used the still-face Paradigm is [10]. Using the CSIRO head tracker [7] they track the head movement in six degrees of freedom (up, down, left, right, backwards, and forwards). The head angles have been converted to angular displacement, acceleration and velocity. Which are then used to check the correlation between affect and head movement. The results suggest that head and facial movement are related to expressing affect of infants and that they are related to the observers ratings.
In [10], affect of the infant by tracking the infants head and facial movement is classified. This is done using two experimental paradigms leading to positive and negative affect. They compared ZFace [11] and the CSIRO head tracker for the tracking of 49 facial landmarks in 3D and the head movement in pitch, roll, and yaw. They find that during negative affect the velocity and acceleration of head and facial movement are greater than during positive affect.
In [19], emotions of infants are classified using the audio of their cry and their facial expressions. The following ‘negative’ emotions are classified: anger, hunger, sadness, pain and fear. This is done by classifying the images and audio separately and merging the decisions from both. The images are classified based on the states of the mouth, eyes, and eyebrows (are they open/closed or up/down). The sound is classified by extracting and analyzing the fundamental frequency and the first three formants of the infants vocalization. Overall they show that fusing both methods leads to a reduction of misclassification.
In [2] classification is done in real-time using facial features and physiological responses. Facial features have been obtained by tracking facial points using the NEVEN vision toolbox. The physiological responses are recorded by multiple sensors, mounted on the subject’s body. Every second of the recordings have been encoded by trained coders using a linear scale, which are converted to a continuous scale using linear regression and neural networks. Chi-square feature selection has been used to extract features from the sensors and footage. For the classification of emotions they used a Support Vector Machine classifier with a linear kernel and Logitboost with a weak decision stump classifier from the WEKA machine learning package. They show that facial features lead to better results than the physiological responses but perform worse when compared against the two together.
The studies above are related with this thesis because they have overlapping fields similar fields such as computer vision with infants, automated classification of infant behaviour, and classification based on facial features. This thesis focuses on automated classification of the still-face paradigm using a neural network with the optical flow.

1.3 Thesis organization

Chapter 2 contains the problem domain, the research question and the expected challenges. In Chapter 3 the used methods are explained and Chapter 4 specifies the dataset. Chapter 5 contains the results of the experiments, finally Chapter 6 summarizes this thesis, gives recommendations for further research and recommends what social scientists can do in the future for automated support of the sfp.

Chapter 2

Problem Definition

First in Section 2.1 the still-face paradigm is explained as well its coding. The research question is posed in Section 2.2. Potential challenges are discussed in Section 2.3.

2.1 Still-Face Paradigm

The still-face paradigm (sfp) is introduced in [22] and consists of face-to-face interaction between infant and parent. It is mostly used for tracking the purposes of social development and to check how infants respond to unpleasant situations. The paradigm has three phases of 2 minutes long: the baseline, the still-face, and the reunion. Figures 2.1(a) and 2.1(b) show an example of a video and the experimental setup of the original sfp. Each phase is briefly described, after that the classification is discussed. An overview of several studies using the sfp can be found in [16].

The baseline consists of normal face-to-face interaction. During the still-face phase the parent maintains a neutral facial expression and stops responding to the child. However if the infant becomes too upset, the next phase is started. In the reunion phase, the parent stops maintaining the neutral facial expression and continues with face-to-face interaction.

The sfp is recorded at home and the recordings are used for manual classification. Gaze, positive, and negative affect are classified for the infant, where sensitivity, intrusiveness, and positive affect are classified for the parent.

A brief description of the general set-up for recording is as follows: a camera stands slightly diagonal behind the parent aiming directly at the infant. The parent can be seen through a mirror placed on a construction behind the infant. Besides holding the mirror the construction also prevents the baby from being distracted by its surrounding. Figures 2.2(a) and 2.2(b) show the setup and footage respectively.

Each phase is classified separately using a discrete scale from 0 to 3. The higher the value the more intense the observed behaviour. A 0 for gaze means that the infant is unresponsive to the parent, where a value of 3 means that the infant consistently gazes at the parent. A brief overview of the scales used for coding can be found in Table 2.1. This table describes for each score the required observed behaviour. The still phase is not coded for the parent due to the maintaining of the neutral expression.

A manual exists which describes what observed behaviour is required for classification. There is still an interpretation factor for the social scientist; behaviour might be classified differently between different scientists. This can be solved by either doing the classification in pairs or by reviewing the classification. But both increase the overall time required for classifying a video.

Every video is manually classified by a social scientist. This is time consuming since in the most optimal

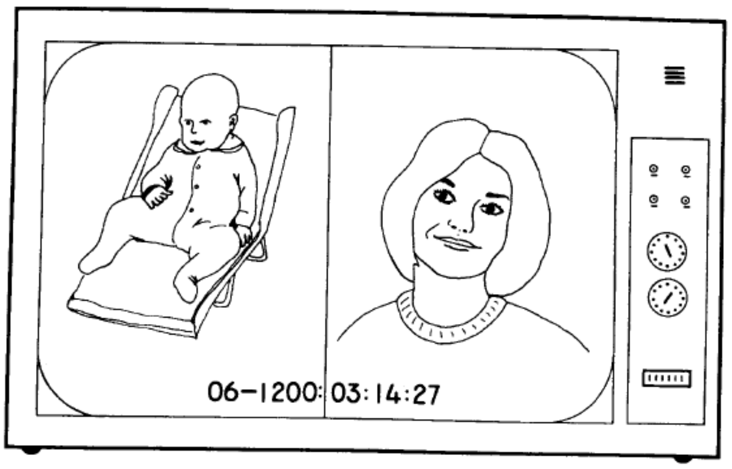


Figure 2
Schema of picture on TV monitor.

(a) Simplified image of the original footage proposed for the sfp [22].

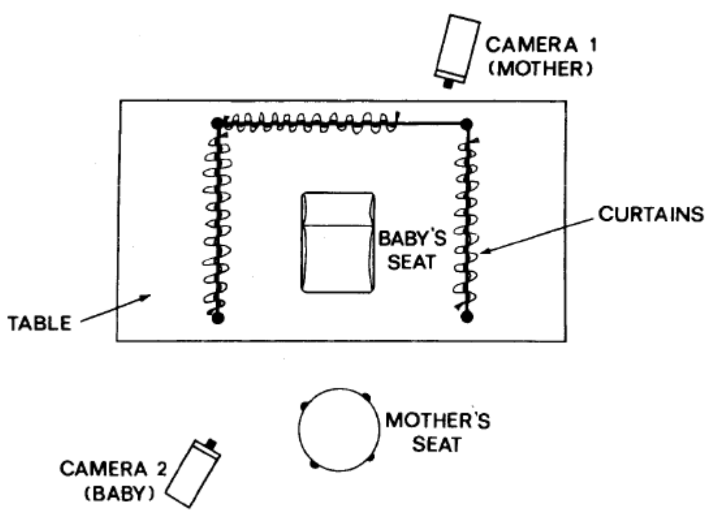


Figure 1
Schema of laboratory during observation of mother-infant interaction. A schematic drawing of the video laboratory illustrating the placement of infant and adult and their respective cameras.

(b) An overview from above from the setup used for the sfp. Image taken from [22].

Figure 2.1: Figure 2.1(a) shows an example of footage of the still-face paradigm. Figure 2.1(b) shows the original experimental setup of the still-face paradigm.

Person	Scale	Score	Description
Parent	Sensitivity	0	No sensitivity
		1	Minimal or low sensitivity
		2	Mixed or moderate sensitivity
		3	Predominantly high sensitivity
	Intrusiveness	0	No intrusiveness
		1	Minimal intrusiveness
		2	Mixed or moderate intrusiveness
		3	Predominantly or high intrusiveness
	Positive Affect	0	No positive affect
		1	Minimal positive affect
		2	Mixed / moderate positive affect
		3	Predominant positive affect
Infant	Positive Affect	0	No positive affect
		1	Minimal positive affect
		2	Mixed / moderate positive affect
		3	Predominant or intense positive affect
	Negative Affect	0	No negative affect
		1	Minimal negative affect
		2	Mixed or moderate negative affect
		3	Predominantly or intense negative affect
	Gaze	0	No gaze
		1	Minimal gaze
		2	Moderate gaze
		3	Predominantly gaze

Table 2.1: A brief overview of the coding manual used for coding the still-face paradigm.

case it takes 6 minutes, and there are over hundreds of videos.

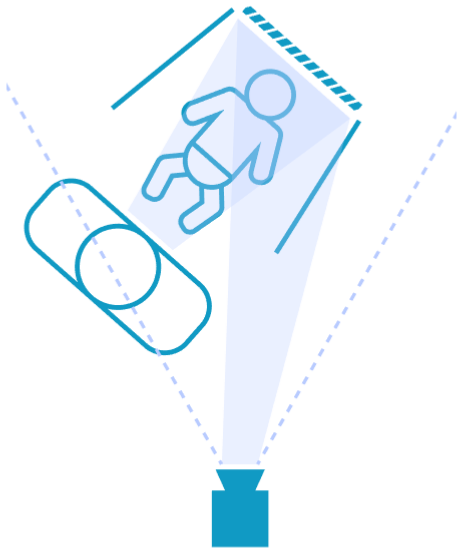
2.2 Problem statement

In this thesis it will be checked whether computer vision can support social scientists with their classification of the sfp. Supporting them should lead to higher consistency between social scientists and it reduces the required time for classification. The following question will be researched: **Is it possible to support social scientists with the automatic classification of home made videos of the still-face paradigm using machine learning and computer vision techniques?**

2.3 Challenges

Here the expected challenges of automated support for the still-face paradigm are explained.

- Difficulties are expected with the concepts of sensitivity and intrusiveness. Intrusiveness is challenging because it is necessary to track the handling of the parent, the corresponding response of the infant, and the response of the parent on that response. This makes it complex to detect. Sensitivity is a very human understanding of behaviour and thus challenging for a computer to detect.
- Since the videos are recorded at home, they are a challenge on their own. Recordings at home causes the videos to have unstable conditions. Challenges that might arise are: the quality of the video, the resolution of the videos, uneven lightning, and obstruction of the lens and or faces.



(a) An overview of the setup used for our recordings.

(b) An edited frame from one of our videos.

Figure 2.2: Figure 2.2(a) shows the setup used in our recordings. The dashed line above the infant stands for the mirror. Figure 2.2(b) shows an edited image (for privacy reasons) from our recordings.

- Obstruction of the lens and or faces are expected, even though the sfp consists of face to face interaction. Some parents interact with their infant using movement or by touch. When getting close with their face, they are no longer visible for the camera and might block the view on the infant as well. It also occurs that arms and or hands block the view of the view by their arms when touching the infant.

- The resolution can also be challenging, as most of the videos have a low resolution (720x576).

- The absence of a baseline is one of the major factors that makes this project challenging. Because the facial gesture of a person is of great importance when classifying emotions. A person that generally frowns in a neutral expression will be classified as upset or mad even though he or she does not feel that way. A baseline gives the system an opportunity to learn about the person it has to classify. Without this, a frowning person will probably be classified as unhappy which may not be true.

- Low contrast gives problems when extracting features. Algorithms have difficulties finding contours of faces with low contrast. This is also the case when there is too much light, which makes it difficult to extract features due to the small differences in colour values.

- Finally, the camera position differs in each video, since it is placed by the observer. It sometimes occurs that the camera is repositioned during the video. This causes the recording angle to be different which might cause problems for a computer to classify the video.

Chapter 3

Methods

Here the used methods are described. To classify a video several steps had to be taken. Every step is explained in a following section.

1. Extract faces for clustering
2. Clustering and splitting up videos
3. Apply Active Appearance Model (AAM) for landmark localization
4. Mask images based on landmarks
5. Stabilize videos, includes rotating and centering of the faces
6. Calculate the optical flow
7. Classification

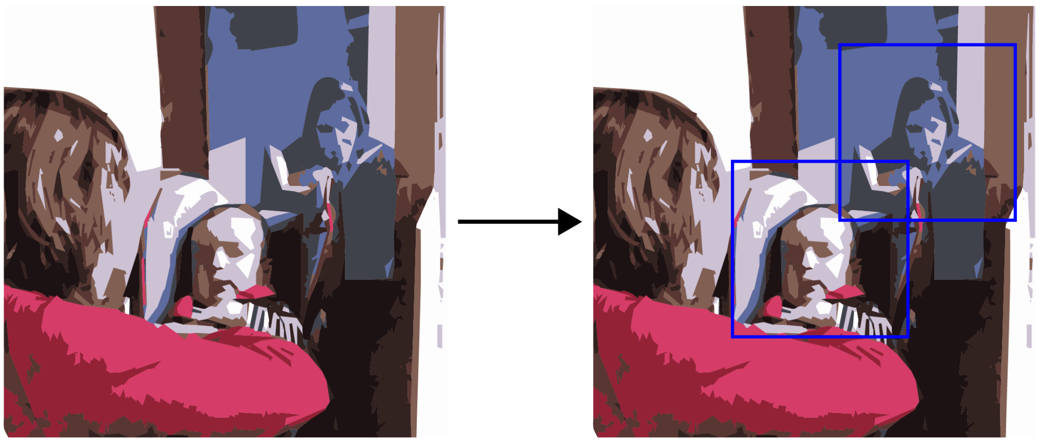
Figures 3.1 and 3.2 summarize these steps, where Figure 3.1 displays steps 1 to 4, and Figure 3.2 shows steps 5 to 7.

3.1 Extracting faces

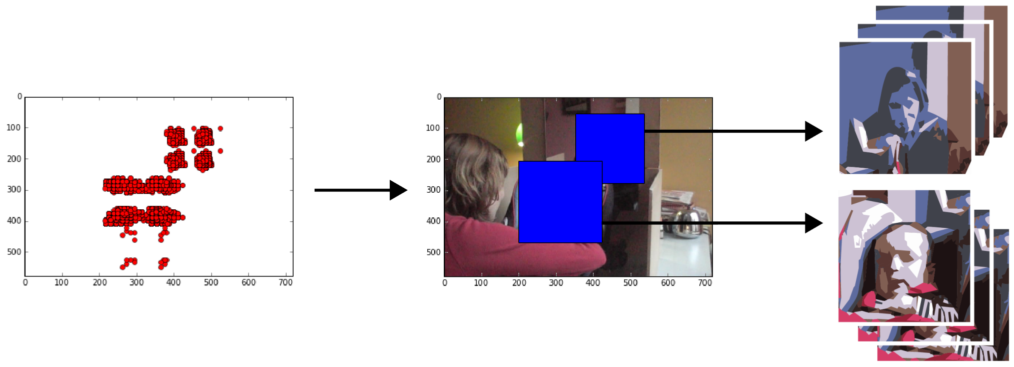
A computer does not automatically know where faces are in a video. They need to be detected. Algorithms have been developed to detect objects. One of these is based on Max-Margin Object Detection (MMOD), proposed in [14] which uses the max-margin approach from [12].
Max-margin requires that every training sample is correctly predicted by a margin, mathematically defined in Equation 3.1. The margin comes from the fact that images are only classified when the prediction of a label is higher than any other prediction. Here x_i stands for the image that needs to be classified, y_i is the label corresponding to x_i and $F(x, y)$ gives the prediction of classifying y given image x.

$$F(x_i, y_i) > \max_{y \neq y_i} F(x_i, y) \tag{3.1}$$

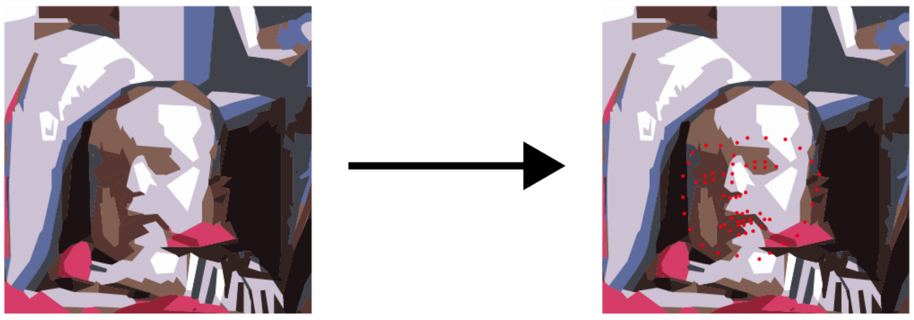
The big advantage of using MMOD over different methods is that it can be used on an entire image and that it does not require a sub-sample of it. This is not necessary because the algorithm does not evaluate



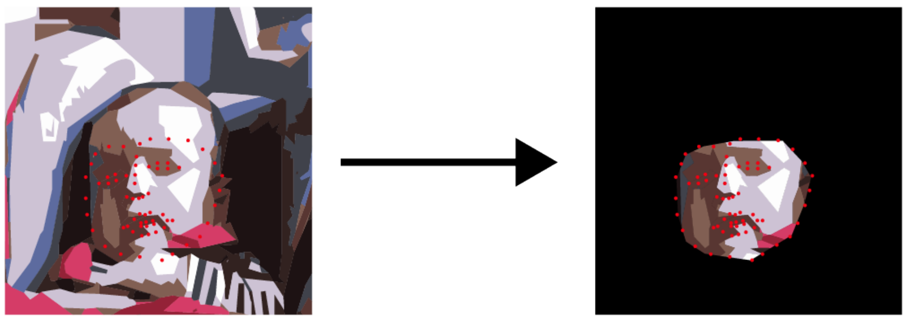
(a) Extracting faces



(b) Clustering and splitting up videos

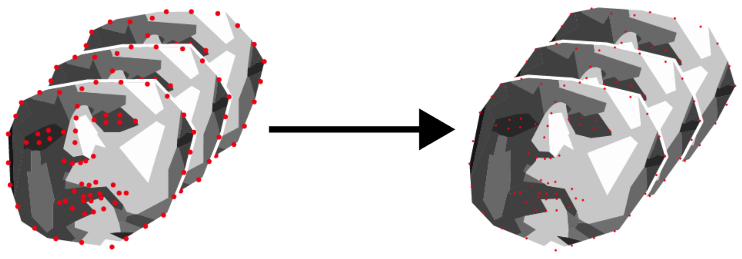


(c) Locate landmarks

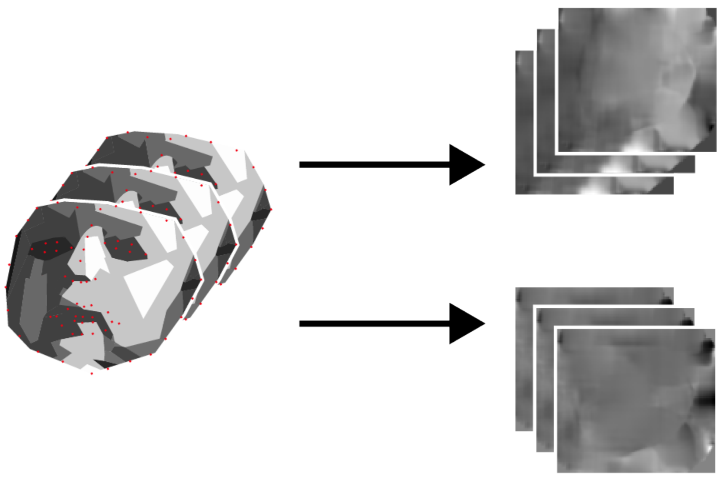


(d) Mask images based on landmarks

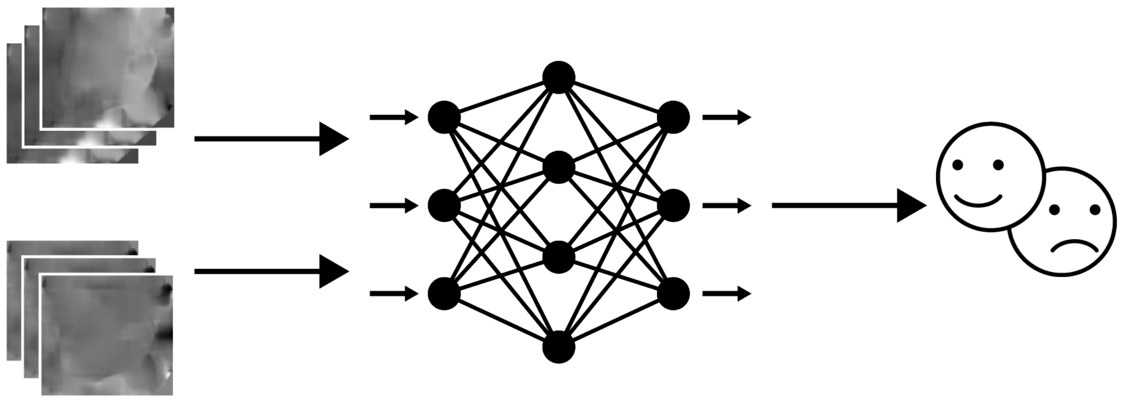
Figure 3.1: The first four steps from the pipeline. Figure 3.1(a) shows the locating of faces in the video. Figure 3.1(b) shows the creation of clusters and splitting up the video using the points from the face detector algorithm.



(a) Align and stabilize images



(b) Calculate optical flow



(c) Classify

Figure 3.2: The final steps of the pipeline. Figure 3.2(a) shows the rotation and alignment of the images. Figure 3.2(b) shows the calculation of the optical flow and finally in Figure 3.2(c) show the classification process using the optical flow and the neural network classifier for classification.

all possibilities. To reduce the number of evaluations the algorithm does not evaluate every possible sub-window from the image, only sub-windows which “do not overlap” are evaluated. Sub-windows are not overlapping when the ratio between the intersection area and total area is less than 0.5:

$$\frac{Area(r_1 \cap r_2)}{Area(r_1 \cup r_2)} < 0.5 \tag{3.2}$$

Here r_1 and r_2 denote two rectangles, and $Area(x)$ defines the number of pixels in x, $\cup$ and $\cap$ are used to denote the overlap and the union of the areas. Besides speeding up the algorithm, evaluation of overlapping windows also provides the opportunity to, up to a certain point, detect overlapping faces.
The detector is applied for each frame of the video. If it detects a face a rectangle containing the face is returned. The coordinates of the corners of this rectangle, $(x_{min}, y_{min}), (x_{max}, y_{min}), (x_{min}, y_{max})$ and (x_{max}, y_{max}), are saved and used for the next step.

3.2 Clustering and splitting up videos

Using the coordinates from the previous step, two clusters (boxes) can be made containing the faces of infant and parent. Everything outside these clusters is discarded to reduce noise.
The clusters are calculated using k-means clustering. A visualization of this algorithm can be found in Figure 3.3. This image shows the result of k-means clustering on two clusters. The algorithm continues running until the position of the centroids does not change substantially, or when a stopping criteria is reached such as a number of iterations.

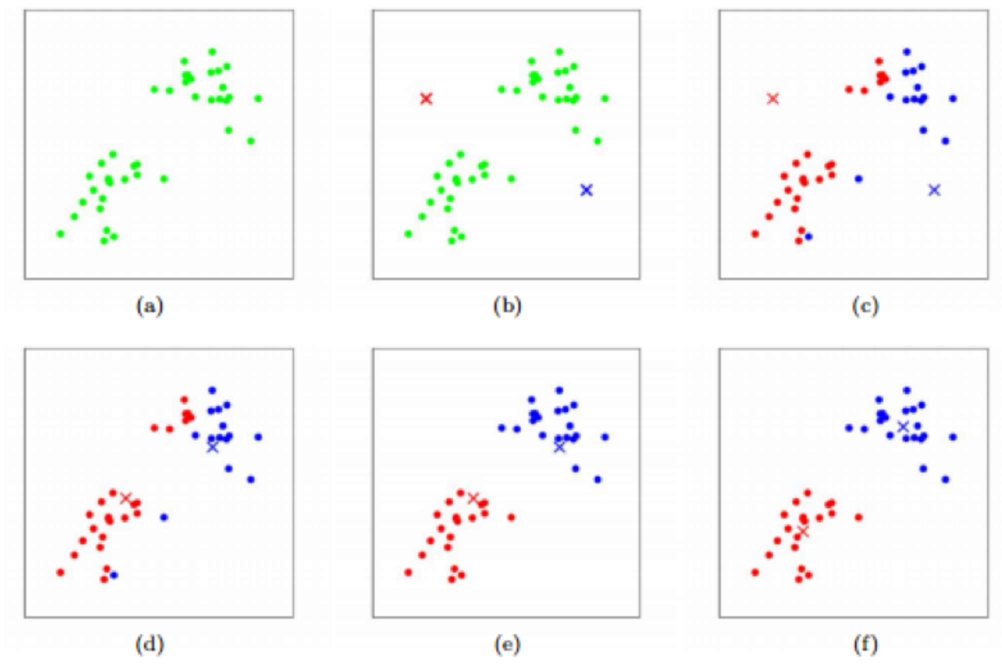


Figure 3.3: The steps of the k-means clustering algorithm. a) displays the initial dataset, b) shows the random initial positions of the centroids of each cluster, c) shows the corresponding assignment of classes to the closest class of the centroid, d) shows the relocation of the centroids such that the distance to all points in that cluster is minimized, e) shows the re-assigment of the labels to the clusters, and f) shows the final result. Image taken from [13].

When the clustering process has completed and every point has been assigned to a cluster, the next

step is the removal of outliers. Outliers are removed to prevent noise and possible detection errors. The process of removing outliers is based on the median distance towards the center, note that the mean is not taken since outliers have a bigger impact on the mean than the median. For each cluster every distance is calculated between its centroid and all points and the median is taken. This median is multiplied with a factor and every point with a distance larger than this value is discarded.
To decide the area of interest from the remaining points the outer most coordinates for each cluster are taken. This results in four values per person in every video: x_{min}, y_{min}, x_{max} and y_{max}. To prevent the possibility that faces are located on the bounds of a cluster, an additional margin is added to each of these points, resulting in the final size of the area of interest. The margin is percentage based on the resolution of the video and can differ for both x and y, ratios of 5 and 10% for the horizontal and vertical margin are recommended.
Using the coordinates, rectangles are created containing an area of interest. These rectangles are extracted and used to create new groups of images containing only a persons face (parent or infant). However, before extraction, additional information is required for labelling: which cluster contains the parent and which one contains the infant, and the start and end times for each phase. The times are also used to determine which frames belong to which phase from the sfp. Frames which do not fall inside these phases are not extracted, these frames are not part of the sfp and thus out of scope.
With the additional information, the areas of interest can now be extracted and saved into new images for each frame. This resulted in five groups of images: two for the parent (baseline and reunion phase) and three (baseline, still-face, and reunion phase).
Each of the following steps is used for each group of images separately.

3.3 Landmark localization and masking

Having a group of images from the previous step, facial landmarks can now be placed. These landmarks are used for masking (discussed here), and alignment (next section). Masking is done to reduce noise when calculating the optical flow.
Since a new group of images is used, it is unknown where the faces are in these images. Therefore the face detector is run an additional time over all images. Its result are used by an Active Appearance Model [5] (AAM), the approach used for the placement of landmarks. AAMs are based on active shape models [6]. They combine models of variation in shape and texture. In this case “texture” refers to the pattern of colours or intensities across an image patch.
The model is trained using a set of annotated images where corresponding points are marked. To train a model for facial images it requires a set of images where the main facial features are marked. The points of the image and model are aligned and each training image is transformed such that it fits the mean shape of the model. The mean model is then scanned into a texture vector which is normalized using a linear transformation. After normalization the texture vector is used to build the texture model. To generate the combined appearance model, the relation between shape and texture is learned. There are different kinds of stopping conditions, i.e. reaching a number of iterations, when the error increases, or when the error does not change substantially.
The model can be fit on unseen images using the following iterative approach:

1. Project the texture sample into the texture model frame
2. Evaluate the new error against the current error
3. Compute the predicted displacement

4. Update the parameters of the model
5. Calculate new points of the model, and recalculate the model texture
6. Sample the image using the new points
7. Calculate the new error
8. Evaluate the error, stop if the stopping condition is satisfied else continue.

When the model is fit on an image, the points from the model are taken as landmarks. Masking is applied by setting the value of every pixel outside the landmarks to 0, a black value. Every pixel not corresponding to the face is now black and will not give results when calculating the optical flow discussed later on.
The result from this steps are groups of masked images with landmarks. Each image is masked such that everything outside the landmarks is black.

Algorithm 1 The steps to determine the shape of the image, where n_x and n_y are the x and y coordinates of the nose, w and h are the height and width of frame f, and F is the collection of all frames from a phase of the spf. The input consist of a series rotated faces with landmarks. The algorithm returns the maximum possible height and width of the provided faces.

Determine window-size for aligned faces

```
$p_{above}, p_{below}, p_{left}, p_{right} \leftarrow 0$
for $f \in F$ do
    if $p_{left} < n_x$ then
        $p_{left} \leftarrow n_x$
    end if
    if $p_{right} < w_f - n_x$ then
        $p_{right} \leftarrow w_f - n_x$
    end if
    if $p_{above} < n_y$ then
        $p_{above} \leftarrow n_y$
    end if
    if $p_{below} < h_f - n_y$ then
        $p_{below} \leftarrow h_f - n_y$
    end if
end for
Height $= p_{below} + p_{above} + 1$
Width $= p_{left} + p_{right} + 1$
```

3.4 Rotation and alignment

The masked images from the previous step now need to be aligned. Alignment results in a more stable optical flow, because facial expressions are easier to calculate when the position of the head is stabilized. Facial landmarks from the previous step are used and are labelled into different facial parts: eyes, eyebrows, mouth, jaw, and nose. For alignment the landmarks labelled ‘’nose” are used, and for rotation the ‘’eye” landmarks are used.
The angle for rotation is based on the eyes, the rotation of the head is calculated using the centre of each eye. The image is then rotated such that the eyes are placed horizontally.
For alignment the nose is used, since it is located near the middle of the face. This makes it a great point for facial alignment. All frames are aligned such that the coordinates of the landmarks on the nose are in the same position.

Calculating the optical flow requires every image to be the same size which might no longer be the case because of the rotation and alignment of the faces. Therefore they need to be transferred into a new frame which is large enough to contain all the aligned faces. Rotation and alignment might have caused every image to be a different size. The size of this frame is determined using Algorithm 1. It works as follows:
The algorithm keeps track of the largest distances towards each edge of the image. The distance towards an edge is calculated using the x and y coordinates of the nose, and they are compared to the previous largest value found. If it is larger, the new value is stored. The final shape of the window is calculated by summing up the respective values and an additional 1, the 1 is added to prevent possible rounding errors.
With the width and height, a new series of images can be created for the aligned faces. Every face is placed such that its nose is located in the middle of the image.
The result of this step is a group of images which are rotated, aligned, and the faces all have their nose in the same spot. Within a group, images now have the same dimensions but they can differ for every video and face.

3.5 Feature evaluation

The features from Section 3.4 are evaluated whether further investigation is required. Here three methods are explained which use all the aligned facial landmarks of a phase in a video. The landmarks are converted into a 2×68 length vector holding both the horizontal and vertical values of the landmarks. In each method these landmarks are converted into a single vector which is used for classification.
All methods are based on the assumption that negative affect lowers the position of the facial landmarks: the positions of the landmarks of the eyebrows and mouth are expected to be positioned lower when negative affect is shown. The first two methods, Sections 3.5.1 and 3.5.2, use real values and all 4 classes for affect. Where the third method in Section 3.5.3 uses a binary approach.

3.5.1 Method 1

The first approach compares two sequential points, p_1 and p_2. If the position of p_2 is higher in the video than the position of p_1 a value of $+1$ is assigned else a value of -1 is assigned. For each point the sum is taken over these values and then they are averaged over the number of frames. Resulting in a value between -1 and $+1$ for a single point in a video. This is defined in Equations 3.3 and 3.4. These values are stored for each phase and finally used to train a classification tree.

$$f(P) = \frac{\sum_{i=0}^{|P|-1} \rho(p_i, p_{i+1})}{|P|} \tag{3.3}$$

Here $f(P)$ calculates the value as input for the classification tree, P holds all values of point p in a phase, $|P|$ denotes the number of points in P, and p_i refers to the i^{th} element of P.

$$\rho(p_1, p_2) = \begin{cases} 1 & \text{if } p_2 - p_1 \geq 0 \\ -1 & \text{if } p_2 - p_1 < 0 \end{cases} \tag{3.4}$$

3.5.2 Method 2

The second approach works in a similar way, here Equations 3.3 and 3.4 are slightly modified resulting in Equations 3.5, 3.6 and 3.7. The main difference is the assignment of the $+1$ and -1 values for each point. The highest and lowest values for each point determine whether $+1$ or -1 is assigned. If the values lies closer to the max-value a $+1$ is assigned else a -1. Again the sum and average are taken which are then used to train a classification tree.

$$f(P) = \frac{\sum_{i=0}^{|P|} \rho(p_i, \alpha(P))}{|P|} \tag{3.5}$$

Again $f(P)$ calculates the value as input for the classification tree, P holds all values of point p in a phase, $|P|$ denotes the number of points in P, and p_i refers to the i^{th} element of P, and β denotes a value of a feature of P and is defined in Equation 3.7.

$$\rho(p, \beta) = \begin{cases} 1 & \text{if } p \geq \beta \\ -1 & \text{if } p < \beta \end{cases} \tag{3.6}$$

Here $\alpha(P)$ is defined as follows:

$$\alpha(P) = \frac{\max(P) - \min(P)}{2} + \min(P) \tag{3.7}$$

3.5.3 Method 3

The third and final approach is different, here not a regular classification tree is used but a binary one. This reduces the number of outputs of the tree from 4 to 2. The class of the video is determined on which score for affect is higher: positive or negative. If positive affect has a higher score than negative a value of $+1$ is assigned, else a 0, if the score for positive and negative are equal $+1$ is assigned. Since this approach uses both scores for positive and negative affect only videos of infants can be used because parents are not coded for negative affect.

The values in the vector undergo the same transformation as in the second method: first Equations 3.5 and 3.6 are applied as with the second method. This is transformed into a binary vector using Equation 3.8, which returns 1 if the value is greater than or equal to 0 and returns 0 if the value is lower than 0.

$$\theta(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \tag{3.8}$$

Assignment of classes is defined in Equation 3.9, here v_{pa} and v_{na} denote the values of positive and negative affect.

$$C(v_{pa}, v_{na}) = \begin{cases} 1 & \text{if } v_{pa} \geq v_{na} \\ 0 & \text{if } v_{pa} < v_{na} \end{cases} \tag{3.9}$$

3.6 Optical flow calculation

The aligned images from the previous step can almost be used by the final neural network classifier. The only remaining objectives are: reshape all images to the same size, and convert the data such that it can easily be used by the neural network.

Different conversion methods exist, but here the focus lies on the optical flow, a visualization of the motion of objects. This is since it can visualize the changes in facial expressions, and can thus be used for the classification of affect.

To calculate the optical flow it requires two frames shortly taken after each other. There are different algorithms to calculate the optical flow. Here the algorithm proposed in [3] is used which is based on [15].

The goal of the optical flow is to find the motion of a pixel between two images. Assume two images I_1 and I_2, and pixel u located on position (x_u, y_u) in I_1. The goal is to find pixel v which is similar to u and located in I_2 on location $(x_u + d_x, y_u + d_y)$. Here vector $d = [d_x, d_y]$ is used to denote the horizontal and vertical displacement of u, also known as the optical flow.

The addition of [15] on this approach is that they use the fact that nearby pixels often have the same optical flow. They propose to use a window when calculating the optical flow. By looking at the neighbouring pixels the results are more stable and precise.

If it is assumed that I_1 and I_2 undergo affine deformation - transformation in translation, rotation, scaling and shearing - in the areas of v and u. Let d_{xx}, d_{xy}, d_{yx} and d_{yy} denote the affine deformation, transformation matrix A can be defined as follows:

$$A = \begin{bmatrix} 1 + d_{xx} & d_{xy} \\ d_{yx} & 1 + d_{yy} \end{bmatrix}$$

The goal is now to find vector d and matrix A which minimize the following function:

$$\epsilon(d, A) = \epsilon(d_x, d_y, d_{xx}, d_{xy}, d_{yx}, d_{yy}) = \sum_{x=-w_x}^{w_x} \sum_{y=-w_y}^{w_y} (I_1(x + u) - I_2(Ax + d + u))^2$$

Here width and height of the window are defined by w_x and w_y. Resulting in a window of size $(2w_x + 1)$ by $(2w_y + 1)$. The main part, $(I_1(x + u) - I_2(Ax + d + u))^2$, calculates the euclidean distance of the displacement of x between I_1 and I_2. This function calculates the displacement of pixels within the window, assuming that the entire window received the same transformation.

To improve the algorithm above, a pyramidal representation is used. This representation creates layers of images, each smaller than the previous one. The advantage of using this, is that it can be used to denote large movement of pixels. Which in turn results in the possibility that large pixel motions (larger than the size of the window) can be handled, which have proven problematic for the algorithm.

The basic approach of the pyramidal implementation is as follows: First the optical flow and the affine transformation are calculated for the lowest level of the pyramid (the smallest image). These results are propagated to the next layer of the pyramid as an initial guess. Using the guess, the actual optical flow and affine transformation are calculated, which are then used as an initial guess for the next layer. This is done for each layer until the optical flow for the original image is calculated.

The optical flow returns both the vertical and horizontal displacement for each pixel. Both values are stored separately as a horizontal and vertical flow. Each flow calculated over all images in a group, and their results are stored in a single file for every group of images. Finally the flows are used as input for the neural network classifier discussed in the next section.

3.7 Neural network classification

Finally classification can be done using the optical flow from the previous step. To classify positive and negative affect, a neural network has been used. Both the horizontal and vertical optical flow are used as input. For each aspect (positive affect infant, negative affect infant, and positive affect parent) a different network is trained.

Neural networks have proven themselves to be a great approach when classifying non-linear problems. Different types exist but here the focus lies on a convolutional neural network (cnn). These networks have proven themselves effective and have been used in several computer vision applications.

A cnn consists of three main components: input layers, hidden layers, and output layers. Figure 3.7 shows an example of a 4 layered cnn, where the layers are coloured green, blue and red for the input, hidden and output layers respectively. As one can see a layer consists of a group of nodes which are connected to other layers, these connections are weighted. Some networks support the possibility that layers are connected with themselves. This is not discussed here since it has not been used and increases the complexity of the network. The depth of a network is denoted by its number of layers.

Input layers are the input of the network; data is inserted into the network through these layers. The range of input values can vary but often they range from -1 to 1. The values from the optical flow have been normalized to these values. The shape of a layer can also have multiple dimensions. They are usually connected to a hidden layer, but it can occur they they are directly connected to the output layer.

Hidden layers are the "unseen" layers of the network, which means that they are usually not visible to the user. Different types exists, but here the focus is on dense and dropout layers. A dense layer (also known as a fully connected layer) is a layer where every node is connected with every node in previous and the following layer. These are the hidden layers in Figure 3.7. The dropout layer has a probability to set the output of a node to zero, resulting in a inactive node and thus having no impact on the result. Dropout is only applied during training to prevent overfitting. The last hidden layer is connected to the output layer.

Finally there are the output layers. These layers return the output of the network. When multiple nodes are used, the node which returns the highest value is often taken as the final output from the network. It is also possible that the output is used as input for a different network.

The main process of a neural network is explained. Every node in a network, has an activation function. This function defines the output of a node given its inputs. Different types exist, but here the focus lies on, sigmoid and softmax, which have shown good results in computer vision applications. Sigmoid is the activation function where softmax is used to classify the output of the network. They are defined as follows:

$$S(x) = \frac{1}{1+e^{-x}}$$
$$\sigma(x)_j = \frac{e^{x_j}}{\sum_{k=1}^{K} e^{x_k}} \text{for } j = 1, \ldots, K$$

With the sigmoid function, $S(x)$, x denotes the input of the node. Where with softmax, $\sigma(x)$, x denotes a vector resembling the output of the network, which is then used for classification of the class. The softmax function requires that all inputs must be real values in the range $[0, 1]$, adding up to 1. Using the softmax activation function the output of node j is calculated as follows:

$$O(j) = S(\sum_{n \in N} O(n) w_{nj})$$

Here N denotes all the nodes from the previous layer, $O(n)$ defines the output from node n, and w_{nj} denotes the weight of the connection between nodes n and j. The final output of the network is calculated by forwarding the values through the subsequent layers in the network, where the output is inserted into the softmax function for the classification of the network.

The network can be trained by adjusting the weights of the connections. This can be done with backpropagation, an algorithm that strives for weights that minimize the total error of the network. Backpropagation consists of two passes: (1) a forward pass in which one example is forwarded through the network and

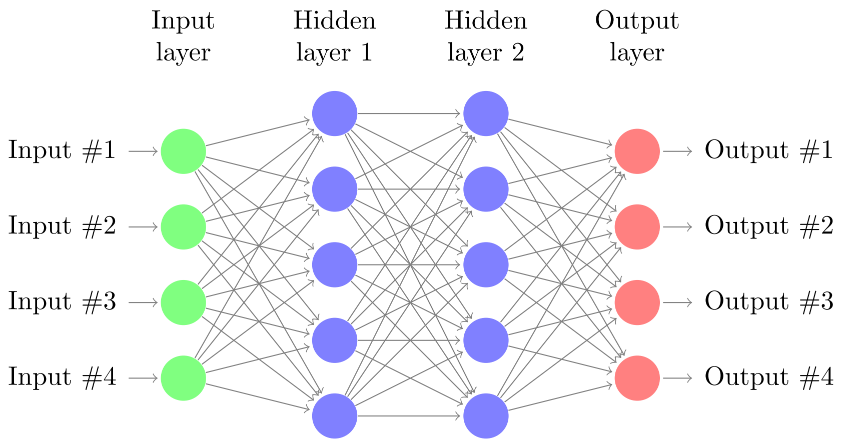


Figure 3.4: An example of a neural network with one input layer, two hidden layers, and one output layer. Here the green nodes are the input nodes, the blue ones are the hidden nodes and the red nodes are the output nodes.

the error of each node and the output layer is computed, and (2) a backward pass which updates the weights using the generalized delta rule. The backwards pass starts at the output layer and the error is propagated through the network layer by layer. In the end every weight is updated. The network's output should become better as more examples are processed. Backpropagation can be done after each sample or in batches. Resultwise there is no difference between the two.
The generalized delta rule is defined as follows:

$$\Delta w_{ji} = \eta \delta_j O(i)$$

$$\delta_j = \begin{cases} \varphi'(v_j)(d_j - O(j)) & \text{If } j \text{ is an output node} \\ \varphi'(v_j) \sum_{k \in K} \delta_k w_{kj} & \text{If } j \text{ is a hidden node} \end{cases}$$

Here w_{ji} denotes the weight from node j to i (from output to input), K are all nodes from the next layer, d_j is the the target from node j, η denotes the learning rate a value which determines the speed of learning, $O(j)$ is the output of j, v_j denotes the activation of j, and φ' denotes the derivative of the activation function. The derivative is taken to determine the step which minimizes the error.
To validate whether training was successful, unseen samples, from which their label is known, are inserted into the network and classified. If the output of the network corresponds with the correct label, the network can be used for classification. If the output does not correspond with the label, the network needs to be configured. Configuring the network can consist of: retrain the network, changing the layout, and adjusting the parameters.
The result of this step is a neural network that can classify the optical flow files from the previous step.

Chapter 4

Dataset

In this section the dataset is described. Section 4.1 contains information about the videos which have been used. In Section 4.2 statistics about the coding of the sfp can be found. Finally section 4.3 gives more information about the files used for the optical flow.

4.1 Videos

The video dataset consists of 43 recordings. The six month old infants, which vary in gender, are recorded during the still-face paradigm (sfp). These recordings are taken at home which leads to unwanted situations such as irregular lightning, occluded lenses and low contrast. Videos in which 30% of the frames did not contain a detectable face have been discarded, leaving a total of 34 videos.

The videos have been recorded at 25 coloured frames per second at a resolution of 720x576. Audio is included with the videos, but not used in this thesis. Information about the duration of each phase can be found in Table 4.1.

4.2 Sfp coding

Every video has been coded according to the coding manual. This manual describes the required observed behaviour for classification. A simplified version of the coding manual can be found in Table 2.1. A summary of the coding values regarding postive and negative affect of the infant can be found in Tables Tables 4.2, and 4.3. Table 4.4 summarizes the coding values for positive affect of the parent. As can be seen, a class balancing problem exists, i.e, Table 4.2 contains little values of 3, and none during baseline.

Phase	Shortest (s)	Average (s)	Longest (s)
Baseline	112	126	177
Still-face	51	111	146
Reunion	75	118	138
Overall	51	118	177

Table 4.1: Details about the video length in seconds, for each phase the shortest, longest and the average are shown.

phase \ class	0	1	2	3	Total
Baseline	28	4	1	0	33
Still-face	22	3	6	2	33
Reunion	20	6	4	3	33
Combined	70	13	11	5	99

Table 4.2: An overview of the different classes for the videos of negative affect for the infant. The values are the number of videos which have been coded with that value.

phase \ class	0	1	2	3	Total
Baseline	1	12	12	8	33
Still-face	20	12	1	0	33
Reunion	5	13	6	9	33
Combined	26	37	19	17	99

Table 4.3: An overview of the different classes for the videos of positive affect for the infant. The values are the number of videos which have been coded with that value.

phase \ class	0	1	2	3	Total
Baseline	0	8	14	10	32
Reunion	2	11	12	6	31
Combined	2	19	26	16	63

Table 4.4: An overview of the different classes for the videos of postive affect for the parent. The values are the number of videos which have been coded with that value.

4.3 Optical flow

Using the methods proposed in Chapter 3 the horizontal and vertical optical flow is calculated and saved. The optical flows are calculated over all frames of a video. However, if both the horizontal and vertical flow of two sequential frames do not contain any values, one of them is discarded.
The 34 videos resulted in a total of 344 files containing the results from calculating the optical flow. The files have a resolution of 110×110 These files took approximately 66GB of diskspace. This is much more than the MPEG-compressed videos, as the optical flow is stored using floats.

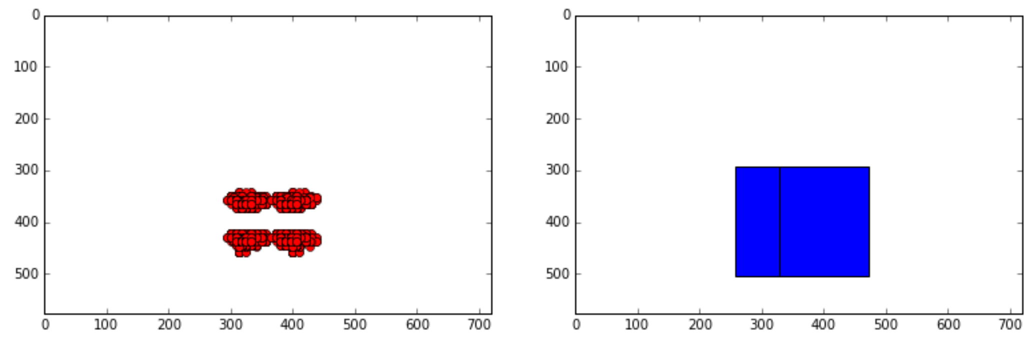
Chapter 5

Experiments and results

First the implementation is discussed in Section 5.1, then in the remainder of this chapter parameters and results are given for each method discussed in Chapter 3.

5.1 Implementation

All programming is done in Python v2.7 using the following packages: Menpo v0.7.2 [1], OpenCV v3.1.0 [4], and Lasagna v0.2 [9] with Theano v0.9.0 [21] as back-end. Both the dlib facial detector and AAM are implemented in Menpo (packages menpodetect v0.4.0 and menpocli v0.4.1), these implementations have been used. To shorten the required time for training the network, calculations have been executed on the GPU using CUDA [18]. For the evaluation of features Scikit-learn [20] v0.18.1 has been used.

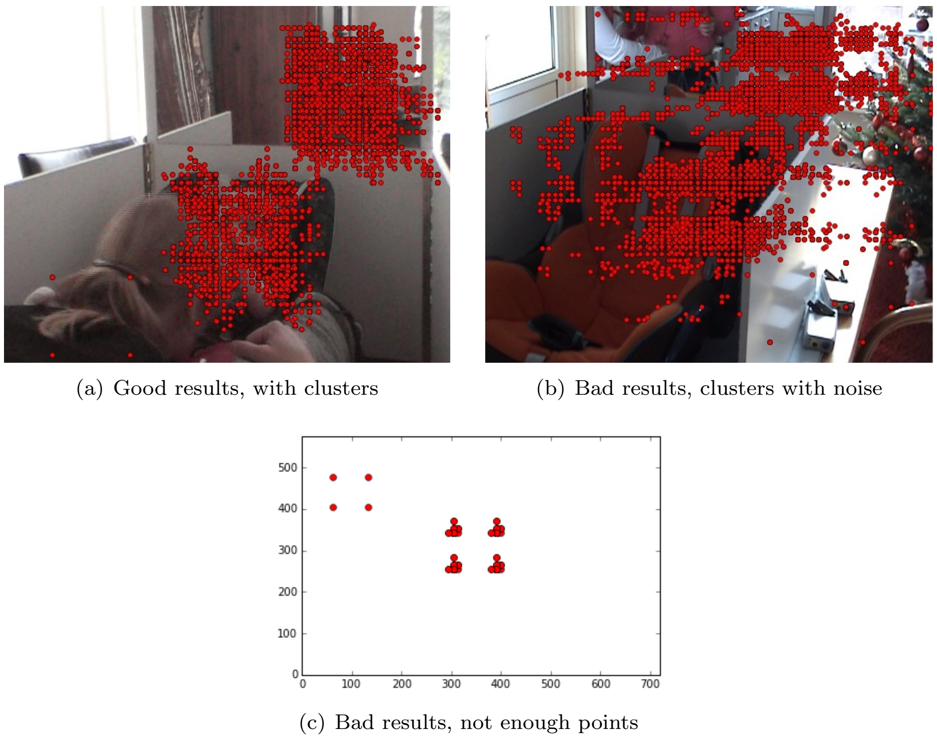


(a) Detecting one face (b) Corresponding boxes

Figure 5.1: Results from the facial detector when one face can only be detected are shown in Figure 5.1(a). Figure 5.1(b) shows the automated clusters created from these points.

5.2 Face extraction and clustering

Here the results are given from the methods described in Section 3.1 and 3.2.
The results of the face detector can be use most of the time. It does not always give the best results mostly because of video footage. Examples of this are provided in Figures 5.2(a), 5.2(b), and 5.2(c). Figures 5.2(a) and 5.3(a) show that enough faces can be detected. However, Figures 5.2(b) and 5.2(c) show that detections can consist of outliers and that not every video contains a lot of detectable faces. As discussed earlier these videos have been discarded.


(a) Good results, with clusters

(b) Bad results, clusters with noise

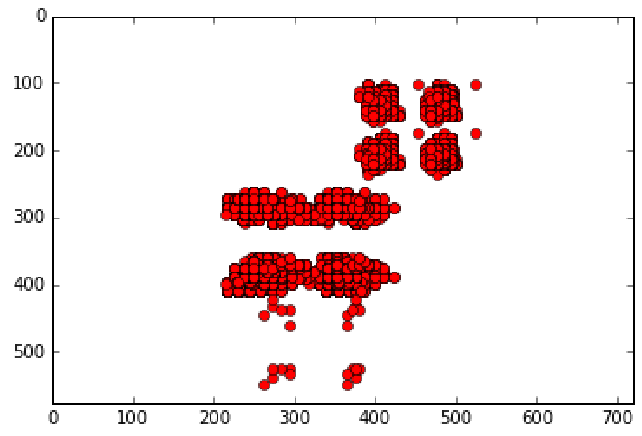
(c) Bad results, not enough points

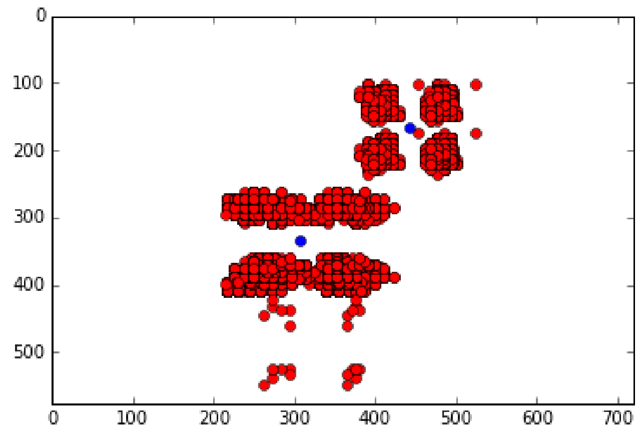
Figure 5.2: Different result by the facial detector on the videos. Figures 5.2(a) and 5.2(b) show that distinct faces can be detected, where Figure 5.2(c) does not. For privacy reasons only a plot is shown in Figure 5.2(c).

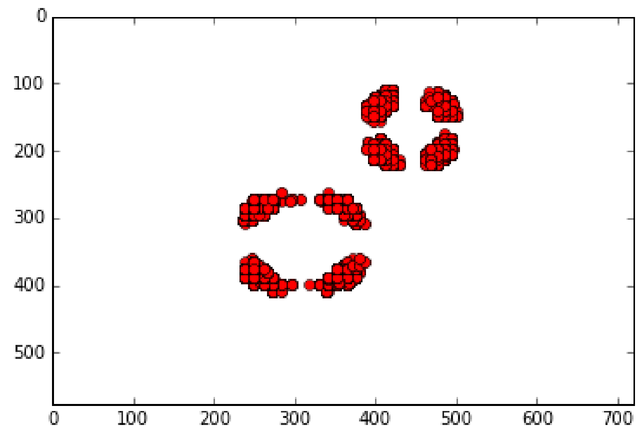
For clustering, a horizontal margin of 5% and a vertical margin of 10% have been used. With a resolution of 720 × 576 pixels, these margins result in an increased width and height of 72 and 115 pixels. The median multiplier, used to remove outliers, has been set at 2.3. Figures 5.3(b) and 5.3(c) show the results before and after removal of outliers.

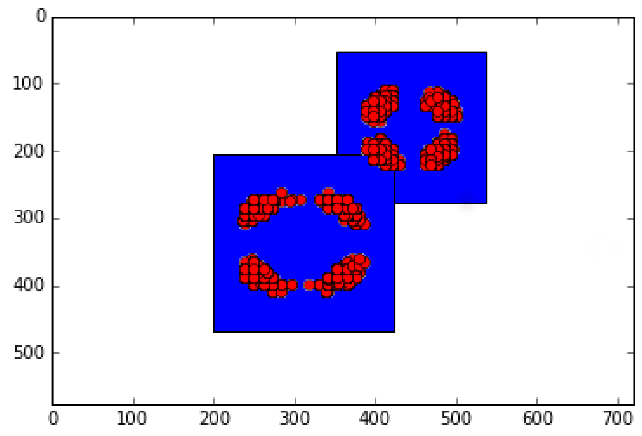
Errors can occur while clustering; when one of the faces is not detected by the facial detector or when one face is detected much more. An example of this can be found in Figures 5.1(a) and 5.1(b), which show an example where only one of the faces is detected. During clustering the assumption is made that there are two clusters in the data. This assumption causes a single cluster to be split in two. The error can be solved by merging the clusters or by lowering the assumption for the video.

Using the provided start and ending times of each phase the clusters are extracted and saved. Without errors this results in a total of 5 clusters.


(a) Results face detector


(b) Centers of the clusters


(c) Remaining datapoints after removing outliers


(d) The boxes generated from the remaining datapoints

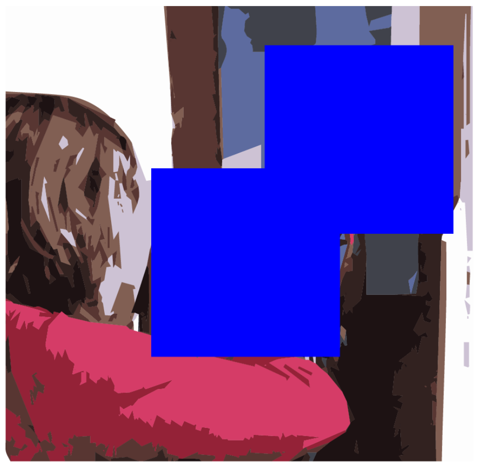

(e) Overview boxes in footage

Figure 5.3: Images that show the steps to create clusters, using the results of the facial detector. The axis show the locations of the pixels in the video. Figures 5.3(a) and 5.3(b) show facial datapoints in red as well as the centers of the clusters in blue. Figure 5.3(c) shows the the remaining points after the removal of outliers. Figure 5.3(d) shows the generated boxes and the remaining points. Finally Figure 5.3(e) shows the boxes displayed over the video.

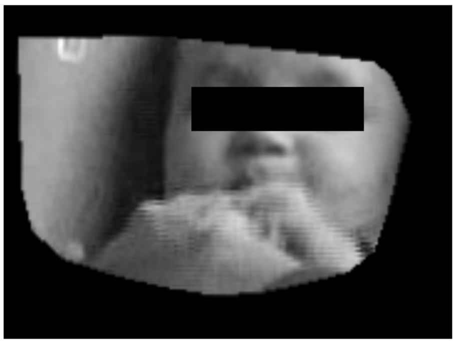
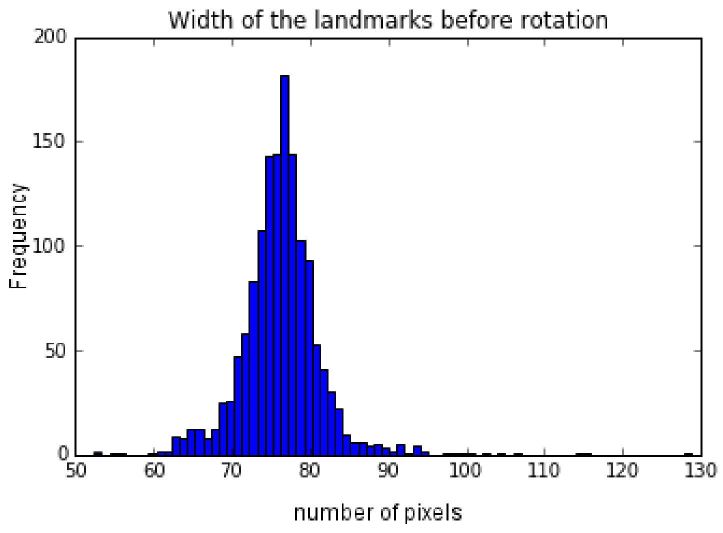
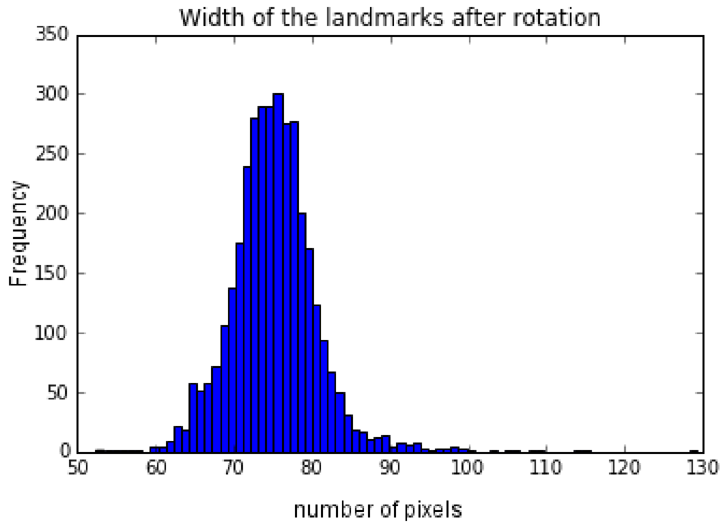

[redacted]

Figure 5.4: An example where the facial landmarks are not placed correctly by the AAM. This gives a wrong optical flow later on in the process.



(a) Distribution before rotation



(b) Distribution after rotation

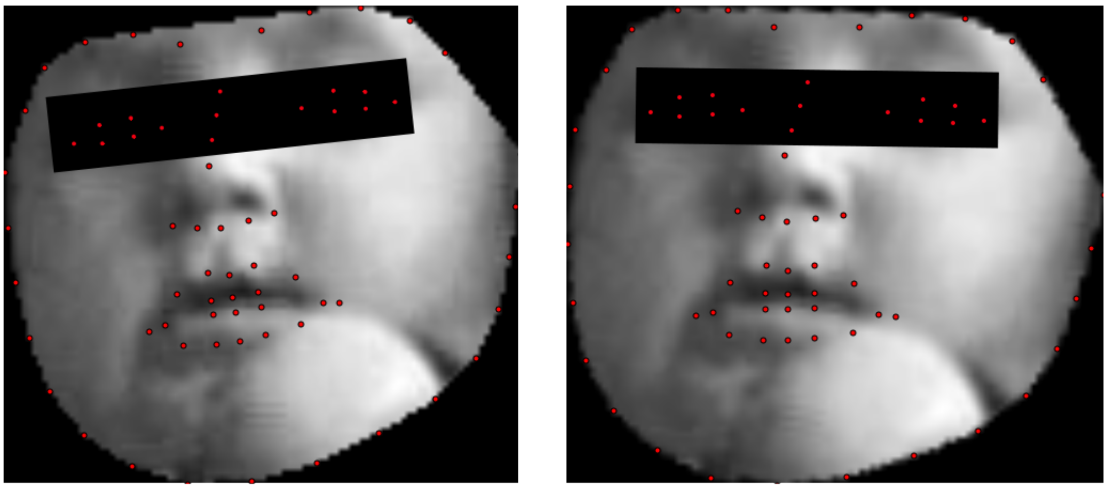
Figure 5.5: Figures 5.5(a) and 5.5(b) show the width of the landmarks before and after rotating, for one video. The horizontal and vertical axes denote the width of the frames and its frequency.

5.3 AAM and masking

Here the results of Section 3.3 can be found.

The AAM is applied to place 68 facial landmarks on the found faces in the images. Landmarks are not always placed correctly, as can be seen in Figure 5.4 which shows a frame with errors after masking. A distribution of the width of the landmarks can be found in Figure 5.5 which shows the width of the landmarks before and after rotation. As can be seen, some outliers exist within the distribution.

These outliers might become a problem further in the pipeline, since they are not detected and propagated to the next step. This might cause corrupted data when aligning the faces or when rescaling the images before calculating optical flow. As described in Section 3.4.



(a) after masking before rotation

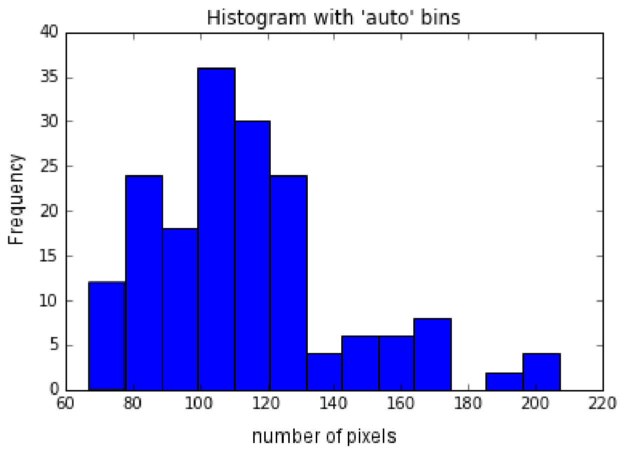
(b) after rotation

Figure 5.6: Figures 5.6(a) and 5.6(b) show an image before and after rotation.

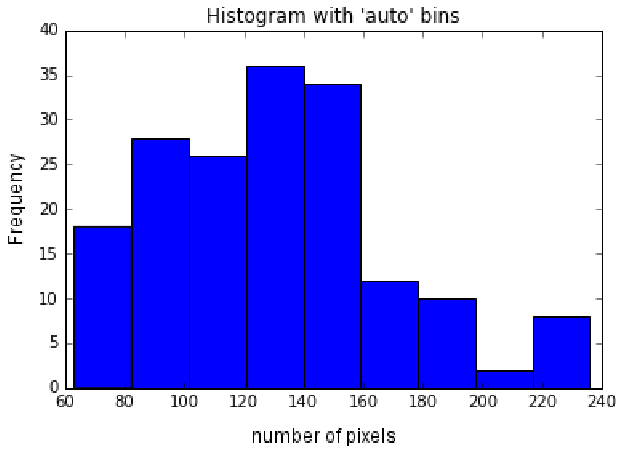
5.4 Stabilizing and centering faces

In this section the results of Section 3.4 are given. The approach for rotating and aligning the faces seems to be good. Results of rotation can be found in Figures 5.6 and 5.8. Figure 5.8 shows in three images the steps taken for rotating an image: the original landmarks, a labelled version of these, and the result after rotation. Figure 5.6 shows the results with an image included. The widths of landmarks before and after rotation can be found in Figures 5.5(a) and 5.5(b) show that rotation reduces the width of the landmarks a little, and that errors stay unchanged.

The widths and heights of the videos after alignment can be found in Figure 5.7. Since the values from the previous step vary, the height and width differ for the videos after aligning the faces. Resizing is described in Section 5.6.

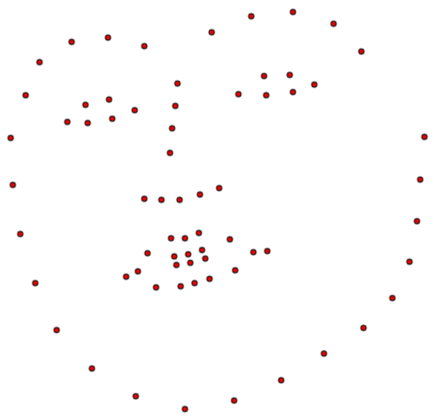


(a) Height images

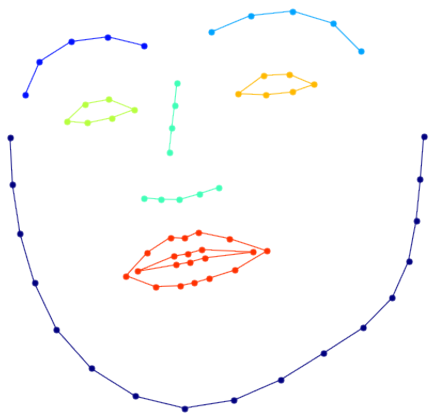


(b) Width images

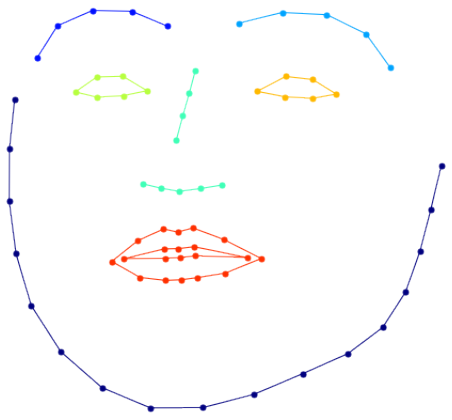
Figure 5.7: Histograms with the height and width of the images for each phase after the alignment step. The horizontal-axis displays the width/height and on the vertical-axis shows the corresponding frequency.



(a) Landmarks



(b) Labelled landmarks



(c) Rotated face

Figure 5.8: Landmarks of a face in Figure 5.8(a) and Figure 5.8(b) shows the labelled version. Here the jaw, left and right eyebrows, left and right eyes, nose and mouth are displayed with different colours. Figure 5.8(c) shows the rotated face from Figure 5.8(b).

Real class	Predicted PA infant	Predicted NA infant	Predicted PA Parent
0	0	1	0
1	0	2	0
2	1	0	0
3	0	3	0

Table 5.1: The results of classifying features using the first method.

Real class	Predicted PA infant	Predicted NA infant	Predicted PA Parent
0	2	1	0
1	0	1	2
2	1	0	0
3	2	2	2

Table 5.2: The results of classifying features using the second method.

5.5 Feature evaluation

Here the results of Section 3.5 are given. For each method all facial landmarks are converted to a 136 length vector, which consists of the horizontal and vertical coordinates. A labeled visualization of the landmarks can be found in Figure 5.10. The labels correspond with the position of the coordinates in the vector, i.e., point 0 consists of the first two values of the vector, a horizontal and vertical one, where point 10 consists of the eleventh and twelfth positions.

A visualization of the values of a single point can be found in Figure 5.9, the red line denotes the α value defined in Equation 3.7 in Section 3.5.2. As can be seen the values vary over approximately 2800 frames.

The classification tree has a depth of 4 unless stated otherwise and it is trained by leaving one sample of each class out; this sample is used for validation. The results for the first method, Section 3.5.1, are shown in Table 5.1. These show that for the infant and parent 3 out of the 12 samples are predicted correctly, resulting in a correct classification rate of 25.0%.

The results of second method, see Section 3.5.2, can be found in Table 5.2. These show that again 2 of the 12 samples are predicted correctly for parent and infant. This results in a correct classification rate of 16.7%.

The results of the third approach, described in Section 3.5.3, are better. As described in Section 3.5 the features are converted to a binary vector and the number of classes have been reduced to 2. A distribution of the classes can be found in Table 5.3. As can be seen approximately 1 of the 5 samples has a higher negative affect than positive.

Confusion matrices of the third approach can be can be found in Tables 5.4, 5.5 and 5.6. Each table shows the results of the entire dataset for classification trees with a different depth. All trees are trained by leaving 3 samples of each class out. The confusion matrices are generated on the entire dataset.

There results of the third approach are notably better than the other ones. With a depth of 2 the correct classification rate is 82.3%, a depth of 3 results in a classification rate of 63.5% and a depth of 4 results in a correct classification rate of 71.9%.

Input class	# classes
0	17
1	79

Table 5.3: The number of classes of infants using the binary input.

		Actual Class		
		True	False	Total
Predicted Class	True	57	4	61
	False	13	22	35
	Total	70	26	96

Table 5.4: The confusion matrix on all samples of the method described in Section 3.5.1. The depth of the tree is 2.

		Actual Class		
		True	False	Total
Predicted Class	True	44	0	44
	False	35	17	52
	Total	79	17	96

Table 5.5: The confusion matrix on all samples of the method described in Section 3.5.1. The depth of the tree is 3.

		Actual Class		
		True	False	Total
Predicted Class	True	52	0	52
	False	27	17	44
	Total	79	17	96

Table 5.6: The confusion matrix on all samples of the method described in Section 3.5.1. The depth of the tree is 4.

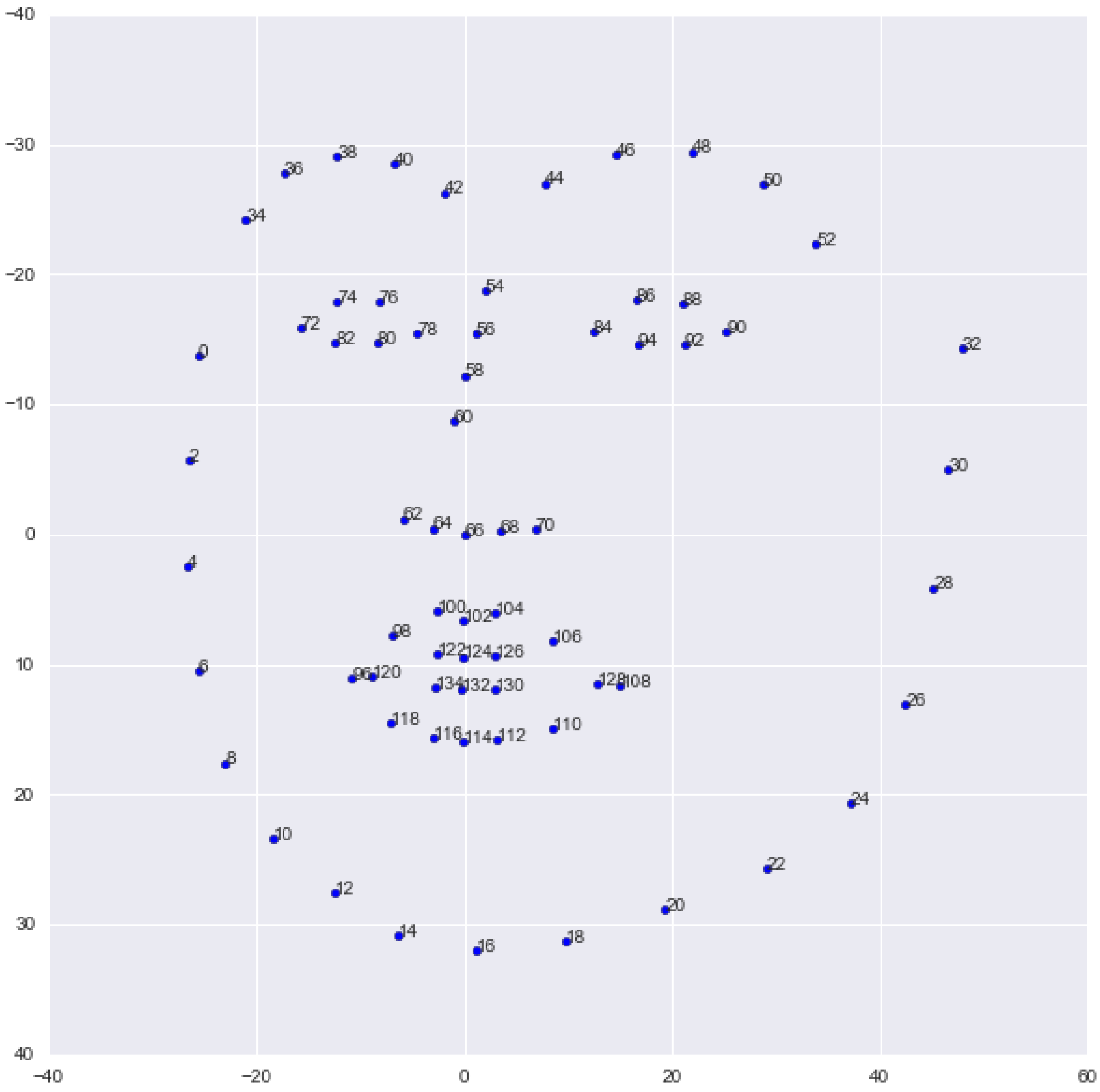


Figure 5.10: Facial landmarks which are annotated corresponding to their order in the vector.

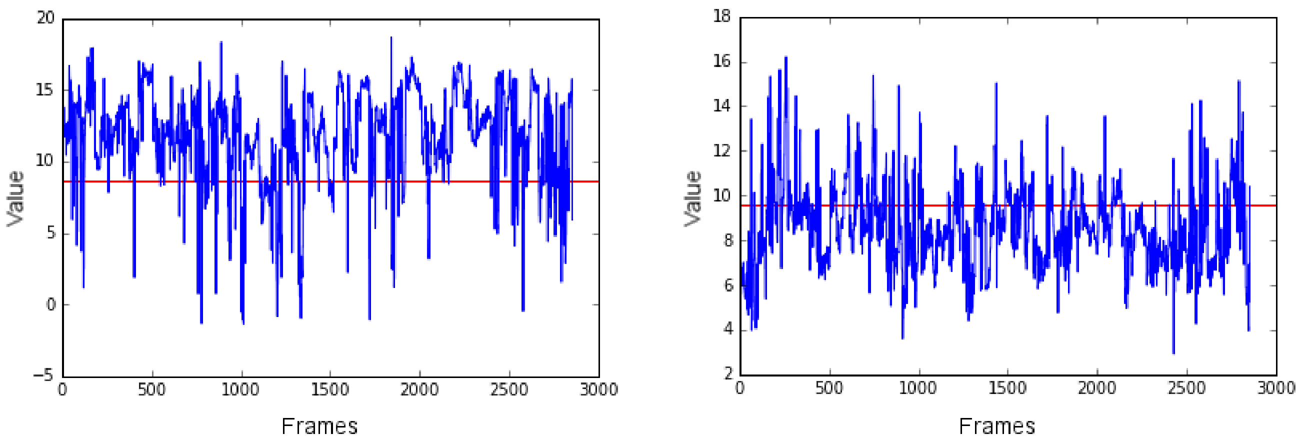


Figure 5.9: The x- and y-values of a point through a phase. The red line denotes the value of $\alpha(P)$ defined in Equation 3.7. The horizontal axis denotes the frame in video and the vertical axis denotes the value of the point.

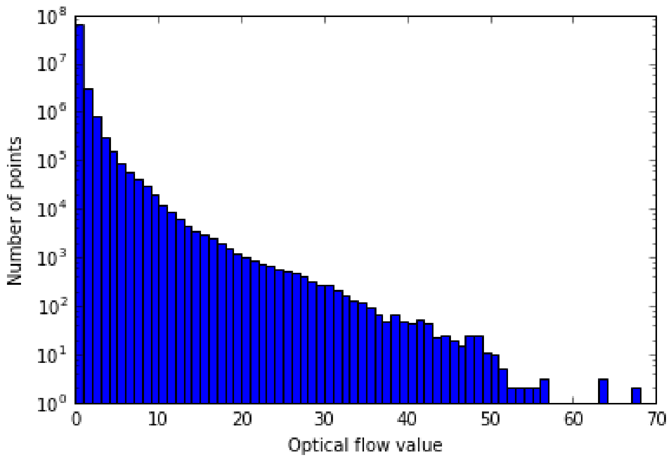


Figure 5.11: The long tail distribution of the optical flow in a video, all values are absolute values.

5.6 Optical flow

This section provides the results of the methods described in Section 3.6. The stabilization step (Section 3.4) causes sizes to differ for each phase and video. Resizing is required to calculate the optical flow. The size of the images is determined on the heights and widths of all images, shown in Figure 5.7. This figure is used to determine the final shape of the images to calculate the optical flow, which is an image of 110×110 pixels.

With these images the optical flow can be calculated. The optical flow is calculated using a window of 15×15 pixels and a 3-level pyramid. A visualization of the optical flow is provided in Figure 5.12 which shows two sequential frames and their corresponding horizontal and vertical optical flows. Note that the values of the optical flow are normalized, causing a dark and light pixels resemble negative and positive values. A value of 0, no flow, consists of grayish pixel.

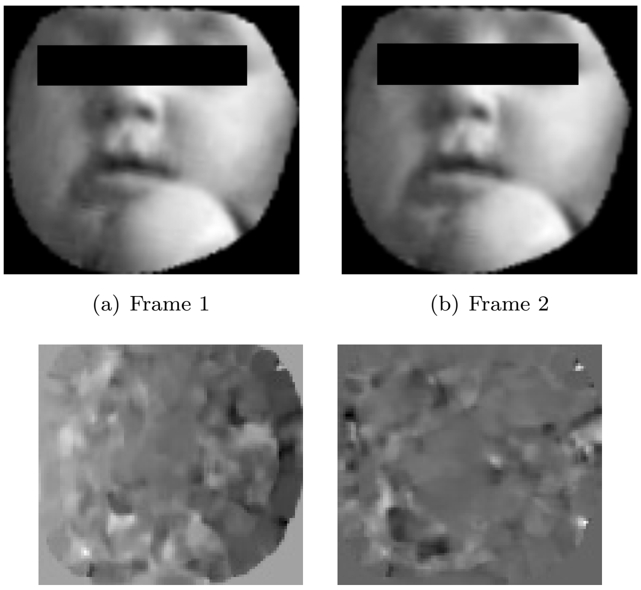
The optical flow values of a video are distributed as a long tail, which can be seen in Figure 5.6. This figure shows the absolute values of the optical flow from an entire video. The outer most 5% values are removed to reduce errors. The huge spike on the left of the value 0 can be explained by the fact that everything outside the mask has an value of 0, resulting in an optical flow of 0.

5.7 Neural network

Here the setup of the neural network is given which is described in Section 3.7 Before the data can be inserted into the network, normalization is applied to range the values between -1 and 1. As can be seen in Figure 5.6 outliers exists. These outliers cause noise when normalizing the data. Therefor the most outer 5% from the data has been removed in the previous step. Resulting in a more evenly balanced input for the network.

The input for our network consists of a 220×110 input; both the horizontal and optical flow. This input is then forwarded through three layers each of 2000 units, and then through two layers of 1000 units, finally an output layer of 4 units is used, one for each class. The Learning rate is set on a value on .01, and momentum is set on .9. The sigmoid activation function is used as the activation function for all nodes, except for the final output of the network where the softmax function is used.

Training of the network is done using the leave one out method; one sample of each class is selected for


[redacted] [redacted]

(a) Frame 1 (b) Frame 2

(c) Horizontal flow (d) Vertical flow

Figure 5.12: A visualization of the horizontal and vertical flow between Figures 5.12(a) and 5.12(b) is displayed in Figures 5.12(c), and 5.12(d). The values are normalized to values between 0 and 255 for visualization purposes, a dark value represents a negative optical flow where a light value represents a positive for the optical flow.

validation, where the remainder is used to train the network.

The class balancing problem that exists within the dataset is solved as follows: only the lowest amount of samples for each class are taken, i.e. if the lowest appearing class has 10 samples, 9 samples of every class are used for training and 1 sample is used for validation. The results of classification are shown in Section 5.8.

5.8 Classification

Here the classification results can be found of the neural network, described in Section 5.7. The network classifies each video to the same class independent of the input. The results of classification on the validation set can be found in in Tables 5.7, 5.8, and 5.9. As displayed the network classifies all classes the same. While training the network, it showed that some frames are classified differently but that the entire video is always classified the same. The classification problem continued to exist through different configurations.

As an alternative for complicated models, the relation between the facial landmarks and affect has been explored as well. This is described in Sections 3.5 and Section 5.5.

Input class	Result classification
0	1
1	1
2	1
3	1

Table 5.7: The results for classifying negative affect for the infant.

Input class	Result classification
0	2
1	2
2	2
3	2

Table 5.8: The results for classifying positive affect for the infant.

Input class	Result classification
0	2
1	2
2	2
3	2

Table 5.9: The results for classifying positive affect for the parent.

Chapter 6

Conclusion

Here the conclusion of this thesis can be found. First in Section 6.1 an answer to the research question is provided: **Is it possible to support social scientists with the automatic classification of home made videos of the still-face paradigm using machine learning and computer vision techniques?** In Section 6.2 recommendations are given for social scientists and finally in Section 6.3 recommendations for future research can be found.

6.1 Conclusion

Here we demonstrated a pipeline for automated support of classification of affect for the still-face paradigm (sfp). State of art computer vision methods have been implemented into a pipeline to reduce the required time by social scientists for classifying this paradigm. The sfp is a paradigm used by social scientists to follow development of interaction between parent and infant. The video recordings of this paradigm are taken at home and contain varying unstable conditions. Classification is done separately for parent and infant and the following aspects are classified: positive and negative affect, sensitivity, intrusiveness and gaze. However, here the focus lies on positive and negative affect.

The pipeline consists of the following steps: face detection, extracting clusters, placement of landmarks, alignment of faces, calculating optical flow, and classification of the flow. Each steps uses the results from the previous step.

Faces can be detected in the videos, only low contrast poses a thread for the detection. The detected faces are then clustered and extracted to reduce noise and to reduce the possibility of faces overlapping each other. Landmarks are placed on the extracted faces using an Active Appearance Model. These landmarks are then used for the alignment, masking and rotating of the faces. When the landmarks are placed, the faces are aligned on the point of the nose, then rotated on the angle between the centers of the eyes, and finally masked. Everything outside the landmarks is masked to disable the calculation of the optical flow later on in the pipeline. The alignment and rotation of faces are done to reduce noise when calculating the optical flow.

After the alignment step, features are extracted from the aligned faces and they are evaluated. Three methods are proposed which are based on the assumption that the position of landmarks are lower when negative affect is shown. These methods use a classification tree and the positions of the facial landmarks of the aligned faces. Two of these methods are used to classify affect individually for each class, as a social scientist would. The other method classifies whether positive affect or negative affect scores higher.

After alignment the optical flow is calculated over the masked images. The optical flow, a representation

of motion between images, is then used as input for a convolution neural network. The pipeline classifies a video based on the output of the network. The results of the pipeline suggest that it is not possible to classify affect for parent and infant using the pipeline. However less complicated methods, such as decision trees, show that classification can be done correctly when using a simpler representation of the data. The approach is also sensitive to errors since there is no validation: this causes errors to propagate through the network when one occurs. An example: an error occurs when placing landmarks causing an increased width. Resulting in a bigger box after alignment, which might cause errors when reshaping the images before calculating the optical flow.
Overall, we do still think that it is possible using our approach to classify negative and positive affect, and ultimately the other aspects of the sfp.

6.2 Recommendations for social scientists

The idea for this research came from a social scientist, therefore recommendations are also given for social scientists. In general, for future research it would be nice to have videos with more stable conditions. Since it is impossible to stabilize all of them suggestions are given which might lead to better recordings which can be used for automated support.

- Try to get the conditions as stable as possible, i.e., do not change the position of the camera while recording.
- Blocking of the faces still occurs, and will never disappear. However, try to position the camera such that blocking the lens by movement of parent or infant is minimized.
- The resolution of the recordings could be higher, these days cameras exist with higher resolutions. More details can be detected with a higher resolution.

6.3 Future research

During this research several methods have been used. Here recommendations are given which might be interesting for future research.

- Different types of algorithms can be tested. The algorithms for the detection of faces, placing landmarks and calculating the optical flow can be replaced with a different algorithm.
- Different types of neural networks can be tested. Here a CNN has been used with softmax and sigmoid activation functions. The layout of the network can also be looked at.
- In our approach face detection has been applied on the entire video. This can be done for each phase separately to have more fitting clusters for each phase.
- A pre-trained AAM has been used, where a custom trained model may lead to better results.
- Audio has not been used, which can be explored.

Bibliography

[1] J. Alabort-i Medina, E. Antonakos, J. Booth, P. Snape, and S. Zafeiriou. Menpo: A comprehensive platform for parametric image alignment and visual deformable models. In *Proceedings of the 22nd ACM International Conference on Multimedia*, pages 679–682. Association for Computation Machinery, 2014.

[2] J. N. Bailenson, E. D. Pontikakis, I. B. Mauss, J. J. Gross, M. E. Jabon, C. A. Hutcherson, C. Nass, and O. John. Real-time classification of evoked emotions using facial feature tracking and physiological responses. *International journal of human-computer studies*, 66(5):303–317, 2008.

[3] J.-Y. Bouguet. Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel Corporation*, 5(1-10):4, 2001.

[4] G. Bradski et al. The opencv library. *Doctor Dobbs Journal*, 25(11):120–126, 2000.

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

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

[7] M. Cox, J. Nuevo-Chiquero, J. Saragih, and S. Lucey. Csiro face analysis. *Brisbane, Australia*, 2013.

[8] A. Dhall, R. Goecke, J. Joshi, M. Wagner, and T. Gedeon. Emotion recognition in the wild challenge 2013. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction*, pages 509–516, 2013.

[9] S. Dieleman, J. Schlter, C. Raffel, E. Olson, S. K. Snderby, et al. Lasagne: First release., Aug. 2015.

[10] Z. Hammal, J. F. Cohn, C. Heike, and M. L. Speltz. Automatic measurement of head and facial movement for analysis and detection of infants positive and negative affect. *Frontiers in ICT*, 2:21, 2015.

[11] L. A. Jeni, J. F. Cohn, and T. Kanade. Dense 3d face alignment from 2d videos in real-time. In *Proceedings of the 11th IEEE International Conference on Automatic Face and Gesture Recognition*, volume 1, pages 1–8. IEEE, 2015.

[12] T. Joachims, T. Hofmann, Y. Yue, and C.-N. Yu. Predicting structured objects with support vector machines. *Communications of the Association for Computation Machinery*, 52(11):97–104, 2009.

[13] KDnuggets. Machine learning algorithms used self driving cars, 2017. [Online; accessed 23-August-2017].

[14] D. E. King. Max-margin object detection. *arXiv preprint arXiv:1502.00046*, 2015.

[15] B. D. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 81, pages 674–679. Vancouver, BC, Canada, 1981.

[16] J. Mesman, M. H. van IJzendoorn, and M. J. Bakermans-Kranenburg. The many faces of the still-face paradigm: A review and meta-analysis. *Developmental Review*, 29(2):120–162, 2009.

[17] N. Nabizadeh and M. Kubat. Brain tumors detection and segmentation in mr images: Gabor wavelet vs. statistical features. *Computers & Electrical Engineering*, 45:286–301, 2015.

[18] J. Nickolls, I. Buck, M. Garland, and K. Skadron. Scalable parallel programming with cuda. *Queue*, 6(2):40–53, Mar. 2008.

[19] P. Pal, A. N. Iyer, and R. E. Yantorno. Emotion detection from infant facial expressions and cries. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages II–II. IEEE, 2006.

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

[21] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016.

[22] E. Tronick, H. Als, L. Adamson, S. Wise, and T. B. Brazelton. The infant's response to entrapment between contradictory messages in face-to-face interaction. *Journal of the American Academy of Child psychiatry*, 17(1):1–13, 1979.

[23] N. Zaker, M. H. Mahoor, W. I. Mattson, D. S. Messinger, and J. F. Cohn. A comparison of alternative classifiers for detecting occurrence and intensity in spontaneous facial expression of infants with their mothers. In *Proceedings of the 10th IEEE International Conference on Automatic Face and Gesture Recognition*, pages 1–6. IEEE, 2013.