# Facial Action Unit Detection using Probabilistic Actively Learned Support Vector Machines on Tracked Facial Point Data

M.F. Valstar, I.Patras and M. Pantic
EEMCS/Mediamatics Department
Delft University of Technology
*{M.F.Valstar, I.Patras, M.Pantic}@ewi.tudelft.nl*

## Abstract

*A system that could enable fast and robust facial expression recognition would have many applications in behavioral science, medicine, security and human-machine interaction. While working toward that goal, we do not attempt to recognize prototypic facial expressions of emotions but analyze subtle changes in facial behavior by recognizing facial muscle action units (AUs, i.e., atomic facial signals) instead. By detecting AUs we can analyse many more facial communicative signals than emotional expressions alone. This paper proposes AU detection by classifying features calculated from tracked fiducial facial points. We use a Particle Filtering tracking scheme using factorized likelihoods and a novel observation model that combines a rigid and a morphologic model. The AUs displayed in a video are classified using Probabilistic Actively Learned Support Vector Machines (PAL-SVM). When tested on 167 videos from the MMI web-based facial expression database, the proposed method achieved very high recognition rates for 16 different AUs. To ascertain data independency we also performed a validation using another benchmark database. When trained on the MMI-Facial expression database and tested on the Cohn-Kanade database, the proposed method achieved a recognition rate of 84% when detecting 9 AUs occurring alone or in combination in input image sequences.*

## 1 Introduction

Humans interact far more naturally with each other than they do with machines. This is why face-to-face interaction cannot be still substituted by human-machine interaction in spite of the theoretical feasibility of such a substitution in numerous professional areas including education and certain medical branches. To approach the naturalness of face-to-face interaction machines should be able to emulate the way humans communicate with each other. Although speech alone is often sufficient for communicating with another person (e.g., in a phone call), non-verbal communicative cues can help to synchronize the dialogue, to signal comprehension or disagreement and to let the dialogue run smoother, with less interruptions. With facial expressions we clarify what is said by means of lip-reading, we stress the importance of the spoken message by means of conversational signals like raising eyebrows, and we signal comprehension, disagreement, boredom and intentions [1]. Machine understanding of facial expressions could revolutionize human-machine interaction and has become, therefore, a hot topic in computer-vision research.

The method proposed in this paper is based on the Facial Action Coding System (FACS) [3]. This is the best known and the most commonly used system developed for human observers to describe facial activity in terms of visually observable facial muscle actions (i.e., action units, AUs). Using FACS, human observers decompose a facial expression into one or more of in total 44 AUs that produced the expression in question.

Previous work on this subject includes automatic detection of 16 AUs from face image sequences using lip tracking, template matching and neural networks [4], color and motion based detection of 20 AUs occurring alone or in combination in profile-view face video [5], detecting 15 AUs occurring alone or in combination by using temporal templates generated from input face video [6] and detection of 18 AUs using wavelets, AdaBoost and Support Vector Machines [7]. For a good overview of the work done in the field see [2, 13].

In this paper we present results of a system designed to detect 16 AUs using features calculated from tracked facial point data. The facial points are tracked using an improved version of Particle Filtering with Factorized Likelihoods (PFFL) proposed in [8]. We extend this particle filtering scheme with a novel observation model combining a rigid and a morphological model. The system uses further a different set of selected features per AU, each of which represents a certain set of spatio temporal relations between the tracked points. AUs are first detected for each frame separately. Then, an adaptive threshold that uses the predictions per frame as input data decides which AUs are

Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)
1063-6919/05 $20.00 © 2005 **IEEE**

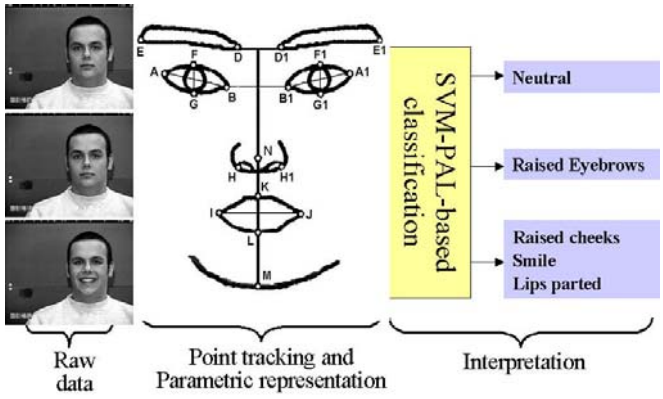


Figure 1: System outline. The 20 fiducial facial points are tracked in raw video data. Features are extracted from the tracked facial points and are used as input for an SVM-PAL based Action Unit classifier.

present in the input video overall. Because of the large number of frames and classifiers to train (16 classifiers and over 15.000 frames), a Probabilistic Active Learning algorithm (PAL) was implemented to reduce the validation time [9]. The system was trained and validated on 167 video's from the MMI-facial expression database [10]. To investigate the generalization ability of the system, we conducted a second validation using all 167 videos from the MMI-facial expression database as the training set and 153 videos from the Cohn-Kanade Face Database [11] as the test set.

The remainder of the paper is organized as follows. In section 2 we describe the process of feature extraction. Section 3 describes the PAL-SVM classification scheme. Next we present the results of our validation study in section 4. Conclusions and suggestions for future research are discussed in section 5.

# 2 Feature extraction

The method proposed here detects activation of AUs by using motion patterns of 20 fiducial facial points (Fig 1) . At this moment the initial positions of these points have to be selected manually in the first frame. The positions in all subsequent frames are determined with a tracker using Particle Filtering with Factorized Likelihoods [8]. Particle Filtering with Factorized Likelihoods is an extension to the Auxiliary Particle Filtering theory introduced by Pitt and Shephard [12], which itself is an extension to classical particle filtering (Condensation) [14].

## 2.1 Condensation

The main idea of particle filtering is to maintain a particle based representation of the *a posteriori* probability $p(\alpha \mid Y)$ of the state $\alpha$ given all the observations *Y* up to the current time instance. This means that the distribution $p(\alpha \mid Y)$ is represented by a set of pairs $\{(s_k, \pi_k)\}$ such that if $s_k$ is chosen with probability equal to $\pi_k$, then it is as if $s_k$ was drawn from $p(\alpha \mid Y)$. In the particle filtering framework our knowledge about the *a posteriori* probability is updated in a recursive way. Suppose that at a previous time instance we have a particle based representation of the density $p(\alpha^- | Y^-)$, that is, we have a collection of $K$ particles and their corresponding weights (i.e. $\left\{\left(s_k^-, \pi_k^-\right)\right\}$). Then, the Condensation Particle Filtering can be summarized as follows:

1. Draw $K$ particles $s_k^-$ from the probability density that is represented by the collection $\left\{\left(s_k^-, \pi_k^-\right)\right\}$.
2. Propagate each particle $s_k^-$ with the transition probability $p(\alpha|\alpha^-)$ in order to arrive at a collection of $K$ particles $s_k$.
3. Compute the weights $\pi_k$ for each particle as follows,

$$\pi_k = p(y \mid s_k) \tag{1}$$

Then normalize so that $\sum_k \pi_k = 1$.

This results in a collection of $K$ particles and their corresponding weights (i.e. $\{(s_k, \pi_k)\}$ which is an approximation of the density $p(\alpha|Y)$.

## 2.2 Factorized Likelihoods

The Condensation algorithm has three major drawbacks. The first drawback is that a large amount of particles that result from sampling from the proposal density $p(\alpha|Y^-)$ might be wasted because they are propagated into areas with small likelihood. The second problem is that the scheme ignores the fact that while a particle $s_k = \langle s_{k1}, s_{k2}, ..., s_{kN} \rangle$ might have low likelihood, it can easily happen that parts of it might be close to the correct solution. Finally, the third problem is that the estimation of the particle weights does not take into account the interdependencies between the different parts of the state $\alpha$.

Particle filtering with factorized likelihoods [8] attempts to solve these problems in one step, given the case that the likelihood can be factorized, that is in the case that $p(y|\alpha) = \prod_i p(y|\alpha_i)$. It uses a proposal distribution $g(\alpha)$ the product of the posteriors of each $\alpha_i$ given the observations, that is $g(\alpha) = \prod_i p(\alpha_i|y)$, from which we draw samples $s_k$. These samples are then assigned weights $\pi_k$, using the same proposal distribution. We now find $\pi_k$ and $s_k$ as follows:

1. Propagate all particles $s_k^-$ via the transition probability $p(\alpha_i|\alpha^-)$ in order to arrive at a collection of $K$ sub-particles $\mu_{ik}$. Note, that while $s_k^-$ has the dimensionality of the state space, the $\mu_{ik}$ have the dimensionality of the partition $i$.

2. Evaluate the likelihood associated with each sub-particle $\mu_{ik}$, that is let $\lambda_{ik} = p(y|\mu_{ik})$.

3. Draw $K$ particles $s_k^-$ from the probability density that is represented by the collection $\{(s_k^-, \lambda_{ik}\pi_k^-)\}$.

4. Propagate each particle $s_k^-$ with the transition probability $p(\alpha_i|\alpha^-)$ in order to arrive at a collection of $K$ sub-particles $s_{ik}$. Note, that $s_{ik}$ has the dimensionality of the partition $i$.

5. Assign a weight $\pi_{ik}$ to each sub particle as follows, $w_{ik} = \frac{p(y|s_{ik})}{\lambda_{ik}}$, $\pi_{ik} = \frac{w_{ik}}{\sum_j w_{ij}}$. With this procedure, we have a particle-based representation for each of the $N$ posteriors $p(\alpha_i \mid y)$. That is, we have $N$ collections $(s_{ik},)\,\pi_{ik}$, one for each $i$.

6. Sample $K$ particles from the proposal function $g(\alpha) = \prod_i p(\alpha_i \mid Y)$. This is approximately equivalent to constructing each particle $s_k = \langle s_{k1}...s_{ki}...s_{kN}\rangle$ by sampling independently each $s_{ik}$ from $p(\alpha_i \mid Y)$.

7. Assign weights $\pi_k$ to the $K$ samples as follows:

$$\pi_k = \frac{p(s_k|Y^-)}{\prod_i p(s_{ik}|Y^-)} \tag{2}$$

The weights are normalized to sum up to one. With this, we end up with a collection $\{(s_k, \pi_k)\}$ that is a particle-based representation of $p(\alpha|Y)$. Note that at the numerator of eq. 2 the interdependencies between the different sub-particles are taken into consideration. On the contrary, at the denominator, the different sub-particles are considered independent. In other words, the re-weighting process of eq. 2 favors particles for which the joint is higher than the product of the marginals.

## 2.3 Rigid and morphologic observation models

In steps 2 and 5 of the PFFL the likelihood and the weight of a sub-particle are determined by applying an observation model. For the system described in this paper we use two different models. Both models are robust color-based observation models for template-based tracking. The first model is suitable for the tracking of rigid motion of the template around a facial micro feature. The second model however, allows for minor morphologic transformations of the template. The models are initialized in the first frame of an image sequence when a set of $N$ windows are centered around the facial micro-features that the user pointed and that will be tracked for the rest of the image sequence. Let us denote with $\boldsymbol{o_i}$ the template feature vector, which contains the RGB color information at window $i$ in frame 1.

We need to define $p(y|\alpha_i)$. Let us denote with $y(\alpha_i)$ the template feature vector that contains the RGB color information at the window around $\alpha_i$. We use a color-based difference between the vectors $\boldsymbol{o_i}$ and $y(\alpha_i)$ that is invariant to global changes in the intensity as follows:

$$c(\boldsymbol{o_i}, y(\alpha_i)) = \left(\frac{\boldsymbol{o_i}}{E\{\boldsymbol{o_i}\}} - \frac{y(\alpha_i)}{E\{y(\alpha_i)\}}\right) \tag{3}$$

where $E\{\boldsymbol{x}\}$ is the (scalar) average intensity on a color template $\boldsymbol{x}$. It is easy to show that the color difference vector $c(\boldsymbol{o_i}, y(\alpha_i))$ is invariant to global changes in the light intensity[1]. Finally, we define the scalar color distance using a robust function $\rho$. Let us denote with $j$ the index to the color difference vector $j$, that is $c_j(\boldsymbol{o_i}, y(\alpha_i))$, the difference in a specific color channel at a specific pixel. The scalar color distance is then defined as:

$$d_c(\boldsymbol{o_i}, y(\alpha_i)) = E_j\{\rho(c_j(\boldsymbol{o_i}, y(\alpha_i)))\} \tag{4}$$

where the robust function that has been used in our experiments is the $L_1$ norm.

The second model allows for non-rigid deformations of the initial template, as mentioned above. Let us denote this unknown transformation with $\phi : N^2 \rightarrow N^2$, a transformation that gives the correspondence between the pixel coordinates of the color template $\boldsymbol{o_i}$ and the image patch $y(\alpha_i)$. Then, let us denote with $y(\alpha_i, \phi)$ the template that results after the nonrigid transformation $\phi$ is applied to the image patch $y(\alpha_i)$. The distance metric $d_m$ for the second model contains two terms: the first term, $d_c(\boldsymbol{o_i}, y(\alpha_i))$, is similar to the distance measure for the rigid observation model, only now we take the minimum color distance over all possible deformations $\phi$. The second term, $d_s(\phi)$, is a measure of the shape deformation that is introduced by the transformation $\phi$. The distance measure is the minimum over all possible transformations, formally:

$$d_m(\boldsymbol{o_i}, y(\alpha_i)) = min_\phi(d_c(\boldsymbol{o_i}, y(\alpha_i, \phi)) + \lambda d_s(\phi)) \tag{5}$$

where the first term is used to penalize large color-based distances, the second term is used to penalize large shape deformations and the parameter $\lambda$ controls the balance between the two terms. Formally, $d_s(\phi)$ is defined as the average Euclidean distance over the pixel based displacements, that is

[1] Note that $c$ contains the color differences over all color channels (R, G, B).

$$d_s(\phi) = E_i\left\{\sqrt{\|i-\phi(i)\|_2}\right\} \quad (6)$$

where $\|x\|_2$is defined as the $L_2$ norm of $x$ and, with a slight abuse of notation, $i$ denotes pixel coordinates. Finally, the observation likelihood reads:

$$p(y|\alpha_i) = \frac{1}{z} exp\left(\frac{-d(y(\alpha_i), \boldsymbol{o_i})}{\sigma_i}\right) \quad (7)$$

where $\sigma_i$ is a scaling parameter and $d(\boldsymbol{x}, \boldsymbol{y})$ is either the distance measure $d_c$ defined in (4) or the distance measure $d_m$ defined in (5) depending on which observation model is applied. $z$ is a normalization term, which in the particle filtering framework can be ignored, since the weights of the particles are renormalized at the end of each iteration so as to sum up to one.

### 2.4 Feature extraction

After tracking $n$ facial micro features in an image sequence containing $l$ frames, we attain a set coordinates $P = \langle \boldsymbol{p_i} \dots \boldsymbol{p_l} \rangle$ with dimensionality $l*n$. In order to extract features that are invariant to rigid head motions within one image sequence we first intra-register all frames within one sequence by subtracting point N (fig 1), which is tracked extremely robust, from all coordinates of facial micro features. Variations in size and locations of the facial micro features between different subjects are minimized by applying a scaling transformation $T$ on the facial points from which we subtract the point $N$ to negate any translational variance. The scaling transformation is applied on the points in a frame. This transformation $T$ is obtained by comparing facial points $B$, $B1$ and $N$ of given subject with their corresponding points in a selected expressionless 'normal' face. Thus, the registered points $\boldsymbol{p'_i}$ are obtained as:

$$\boldsymbol{p'_i} = T(\boldsymbol{p_i} - \boldsymbol{N}) \quad (8)$$

From the set of points $P' = \langle \boldsymbol{p'}_1 ... \boldsymbol{p'}_n \rangle$ we extract for every AU a set of features $F_a$ with dimensionality $l * d_a$. The features we extract for our system are simple relations between the coordinates, based on the rules for AU activation as described in [13]. The relations are listed in Table 1.

Finally, we apply a temporal filter on the value of the features $F_a$ to arrive at a feature set $F'_a = \langle f'_{a1}, f'_{al} \rangle$ that is more robust to noise and reveals the temporal pattern of an AU activation more clearly.

$$f'_{ai} = \frac{1}{7}\sum_{i-3}^{i+3} f_{ai} \quad (9)$$

Figure 2 clearly shows the noise reduction achieved by (9).

Table 1: Feature representation of changes in position of fiducial facial points

| Feature | Features for facial point features |
|---|---|
| $Edist(P_1, P_2)$ | Euclidean distance between the points $P_1$ and $P_2$ |
| $EdistInc(P_1, P_2, n)$ | The Euclidean distance increase between points $P_1$ and $P_2$ at frame $n$ relative to their distance at frame 1 |
| $xDistFromN(P)$ | The vertical distance between point $P$ at frame $n$ and point $P$ at frame 1 |
| $yDistFromN(P)$ | The horizontal distance between point $P$ at frame $n$ and point $P$ at frame 1 |

## 3 SVM Classification

Support Vector Machines (SVMs) have proven to be extremely efficient classifiers, achieving classification rates unparalleled by any other classifier in domains as diverse as marine biology, face detection and speech recognition. They are non-linear, generalize very well and have a well-founded mathematical basis. The essence of SVMs can be summarized in three steps: maximizing the hyperplane margin, mapping the input space to a (hopefully) linearly separable feature space and applying the 'kernel trick' to the results of the first two steps. In the remainder of this paper, $\alpha$ denotes the Lagrange parameters that describe the separating hyperplane in a SVM.

Maximizing the margin of the separating hyperplane $\boldsymbol{w}$ results in a high generalization ability. In words, it is the problem of finding the hyperplane that maximizes the distance between the support vectors (SVs) and $\boldsymbol{w}$. This involves finding the nonzero solutions $\alpha_i$ of the Lagrangian dual problem, which is a quadratic programming problem and can be solved efficiently. Having found the support vector weights $\alpha_i$ and given a labeled training set $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ the decision function in input space is:

$$f(x) = sgn\left(\sum_{i=1}^{m} \alpha_i y_i \langle \boldsymbol{x}, \boldsymbol{x}_i \rangle + b\right) \quad (10)$$

where $b$ is the bias of the hyperplane and $\langle a, b \rangle$ is the inner product of $a$ and $b$. Off course, most real-world problems are not linearly separable in input space. To overcome this problem, we map each input sample $\boldsymbol{x}$ to its representation in feature space $\Phi(\boldsymbol{x})$ in which we can apply our algorithm for finding the maximal margin hyperplane. The

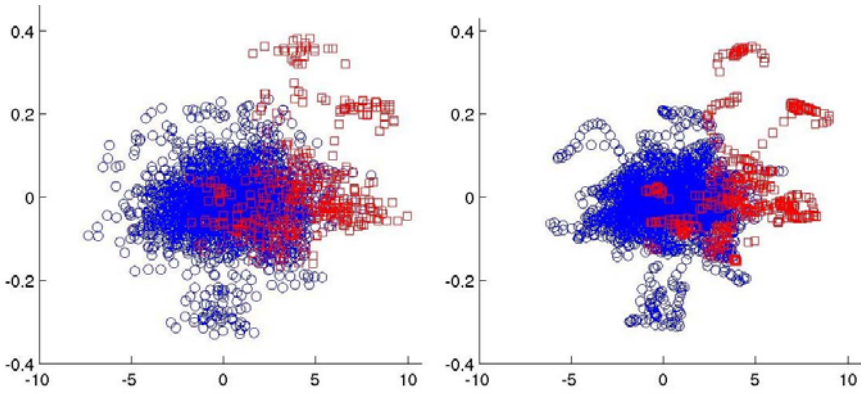


Figure 2: Noise reduction by temporal filtering of two features for detection of Action Unit 25 (lips parted). The x axis represents the feature EdistIncrease(I,J) and the y axis the feature {yDistFromN(L)-yDistFromN(K)} (see Fig 1 for the location of points I, J, K and L). Blue circles are negative samples (lips together), red squares positive examples (lips parted). The left figure shows the unfiltered features, while the right figure clearly shows a reduction in noise and clearer spatio-temporal patterns.

third step is probably the most important step. Maximizing the margin and evaluating the decision function both require the computation of the dot product $\langle \Phi(x), \Phi(x_i) \rangle$ in a high-dimensional space. These expensive calculations are reduced significantly by using a Mercer kernel $K$, such that

$$\langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{x}_i) \rangle = K(\boldsymbol{x}, \boldsymbol{x}_i) \tag{11}$$

The patterns which we want to detect using our maximal margin classifier do not need to coincide with the input $\boldsymbol{x}$, we might as well apply our decision function (10) directly on $\Phi(x)$. Substituting (11) for the inner product, the decision function in feature space directly becomes

$$f(\boldsymbol{x}) = sgn\left(\sum_{i=1}^{m} y_i \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i) + b\right) \tag{12}$$

### 3.1 Probabilistic Active Learning

Because of the large amount of data points used in our validation phase (over 15.000 frames), cross validation becomes an intractable problem. To overcome this problem, we implemented a Probabilistic Active Learning algorithm (PAL) [9]. PAL is computationally efficient when dealing with large sets of data. The algorithm iteratively builds the SV set, using only a small subset of the training samples on which it trains a support vector classifier. The algorithm estimates the likelihood that a new example belongs to the actual support vector set and selects a set of $p$ new points according to this likelihood, which are then used along with the current set of SVs to obtain the new SV set. The likelihood of an example being an SV is estimated using a combination of two factors: the margin of the particular example with respect to the current hyperplane and the degree of confidence that the current set of SVs spans the actual hyperplane (not, as Mitra et al. [9] propose, the actual set of SVs, as the set of SVs spanning the hyperplane does not need to be unique). This confidence factor $c$, which varies adaptively with each iteration, can also be seen as a measure for how close the current hyperplane is to the actual hyperplane. Therefore $c$ can be used as a indication to either pick for the next iteration a high number of new samples close to the current hyperplane (high $c$) or instead far away from the hyperplane (low $c$). So instead of being randomly generated, the new set of samples for each iteration is generated according to a probability $P_{\xi(\boldsymbol{x}, f(\boldsymbol{x}))}$ where $\xi(\boldsymbol{x}, f(\boldsymbol{x}))$ denotes the event that example $\boldsymbol{x}$ is an SV. If $\langle \boldsymbol{w}, b \rangle$ is the current separating hyperplane, we have:

$$P_{\xi(x, f(x))} = \begin{cases} c & if\ y(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b) \leq 1 \\ 1 - c & otherwise \end{cases} \tag{13}$$

Here $c$ is the above mentioned confidence factor. This factor is estimated as follows. Let the current set of SVs be denoted by $S = \{s_1, s_2, ..., s_l\}$. Also, consider an integer $k$ (say, $k = \sqrt{l}$). For every $\boldsymbol{s}_i \in S$, compute the set of $k$ nearest points in the train set $\boldsymbol{x}$. Among the $k$ nearest neighbors, let $k_i^+$ and $k_i^-$ number of points have labels +1 respectively -1. The confidence factor $c$ is then defined as:

$$c = \frac{2}{lk} \sum_{i=1}^{l} min\left(k_i^+, k_i^-\right) \tag{14}$$

Note that the confidence factor varies between zero, when all nearest neighbors have the same label, and, one when the class labels around each support vector are evenly distributed. This results in an adaptive algorithm that starts

with finding the general location of the separating hyperplane and then proceeds with fine tuning the exact location of $\boldsymbol{w}$.

# 4 Experimental evaluation

Until recently, the Cohn-Kanade database was the only benchmark set for research efforts in automating facial expression analysis. However, since this data set exhibits a number of drawbacks, another benchmark facial expression data set has been recently proposed. The pertinent MMI facial expression database contains more than 800 face video sequences recorded in true color instead of gray scale, having a frame rate of 24 instead of 12 frames per second, having no time stamps occluding the facial components and containing a large number of AU-coded videos in a frame by frame manner. Finally, this database has been developed as a web-based direct-manipulation application, allowing easy access and easy search of the available images [10].

To test the performance of our system, we performed two validation studies. For the first study, we applied a leave-one-session-out cross validation using samples of the MMI-Facial expression database. The second study evaluates how well the system generalizes on new data. For this purpose we trained the system using 167 samples from the MMI-Facial expression database and we tested it on 153 videos from the Cohn-Kanade Database. Unfortunately, the Cohn-Kanade database does not contains video samples picturing all AUs that our system can recognize. Hence, we were not able to perform this validation for all 16 AUs used in the first study, but only for 9 of those.

## 4.1 MMI Facial expression database validation

We trained and tested 16 binary classifiers for 16 different AUs using data on 20 facial micro features (fig 1) tracked in 167 videos from the MMI-Facial Expression Database. The data are of 15 different subjects, displaying facial expressions produced on command. The utilized data picture not only the 16 AUs we wish to detect but also other AUs. This way we know that our system will work in real-life situations where people can display any facial expression, although we will not be able to recognize them all. Validation was performed using a leave-one-session-out scheme, where each session is an image sequence picturing one facial expression. Ideally, for training an SVM classifier one should have at least 15 samples of every class one wishes to detect [7]. Considering the small number of samples we have for some AUs (see table 2), any other cross validation technique would result in lower detection rates.

The SVM classifier detects AUs per frame. Since we want to determine the presence of an AU within a video overall, we add a decision layer that adaptively computes a threshold, favoring the recall over the precision of the classification of AUs in facial video. First, we determine a set of AU-activation predictions for each frame in every of the 167 facial videos. Suppose the SVM determined that a video $\boldsymbol{x}$ has $m$ frames where a certain AU is active. Let $\boldsymbol{N}_p$ be a vector containing for every video in which the pertinent AU is activated the number of frames that the SVM predicted to have that AU activated. Similarly, let $\boldsymbol{N}_n$ be the vector containing per video in which the pertinent AU is not present the number of active frames that the SVM falsely predicted that AU is to be present. Let $m_p = \min(\boldsymbol{N_p})$ be the length of the shortest video segment belonging to $\boldsymbol{N_p}$ and let $0 < m_{np} = \max(\boldsymbol{N_n}) < m_p$ be the length of the longest segment belonging to the subset of $\boldsymbol{N_p}$containing video segments with length smaller than $m_p$. The threshold $\theta$ that is used to decide whether the test sample $\boldsymbol{x}$ contains the AU under investigation (i.e. $m > \theta$) is now defined as:

$$\theta = \frac{m_p + m_{np}}{2} \tag{15}$$

The evaluated system uses for every binary classifier a few selected features $F'_a$. While for most AUs it was possible to derive features directly, for AU6 and AU9 (cheek raising and nose wrinkling, respectively) we cannot formulate any features that directly indicate their activation. Therefore, for the detection of AU6 and AU9 we use all features defined for the other 14 AUs. Table 2 shows our results using the leave-one-session-out validation. Columns three to five list the classification rate, recall and precision. The classification rate is the number of correctly classified samples divided by the total number of samples. The recall is defined as the number of positive samples from the ground truth that are correctly classified divided by the total number of ground truth positive samples. Precision is defined as the correctly classified samples divided by the sum of correctly classified positive samples and the number of false positives. As can be seen from table 2, the usage of all features enables us indeed to detect AU6 and AU9, even though we could not define any specific set of features that would directly indicate the activation of these AUs.

An interesting observation is the detection of AU6. AU6 occurs naturally together with AU12 and AU13 (smiles). However, in posed smiles, AU6 is often not activated. In our validation set, 5 out of 21 'smiles' were not accompanied by AU6. Still, instead of learning the correlation with AU12 and AU13 and, in turn, resulting in a large number of false positives, our classifier for AU6 seems to have learned to distinguish between the real and the posed smiles as can be seen from the high precision achieved by the classifier for this AU.

Table 2: Cross validation results on the MMI facial expression database. The second column lists the number of positive/negative sessions for the specified Action Unit. A session is positive for an AU if the AU is contained in that session.

| AU | truth | cl. rate | Recall | Precision |
|---|---|---|---|---|
| 1 | 13/154 | 1.00 | 1.00 | 1.00 |
| 2 | 10/157 | 1.00 | 1.00 | 1.00 |
| 4 | 22/145 | 0.96 | 0.91 | 0.83 |
| 6 | 16/151 | 0.96 | 0.69 | 0.92 |
| 9 | 10/157 | 0.99 | 1.00 | 0.83 |
| 10 | 15/152 | 0.90 | 0.67 | 0.48 |
| 12 | 11/156 | 0.99 | 0.82 | 1.00 |
| 13 | 10/157 | 0.97 | 0.90 | 0.69 |
| 16 | 17/150 | 0.77 | 0.53 | 0.23 |
| 18 | 13/154 | 0.95 | 0.77 | 0.67 |
| 20 | 10/157 | 0.98 | 0.80 | 0.80 |
| 22 | 8/159 | 0.95 | 0.75 | 0.46 |
| 25 | 75/92 | 0.90 | 0.87 | 0.92 |
| 26 | 17/150 | 0.95 | 0.65 | 0.85 |
| 27 | 10/157 | 1.00 | 1.00 | 1.00 |
| 30 | 8/159 | 0.97 | 0.75 | 0.67 |
| | Total: | 0.95 | 0.82 | 0.77 |

### 4.2 Validation using two databases

To determine the generalisation ability of our system we trained it on one database and we tested it on another database. Training was performed using 167 videos from the MMI-Facial expression database while testing has been done using 153 videos from the Cohn-Kanade Database. To do so, the Cohn-Kanade database video samples needed to be AU-coded in the frame by frame manner as well. We did that and achieved the results shown in table 3. Clearly, although the classification rate is somewhat lower than it is the case in the first validation study, the system still performs very well with a 84% overall recognition rate. Most differences in classification rates arose due to the differences of the facial expressions recorded for the two databases. The Cohn-Kanade database contains recordings of multiple-AU facial expressions with sometimes extremely subtle AU activations. In contrast, the set we used from the MMI-Facial expression database has only one or two AUs active for each video and the AU activations are always clearly visible. The problem is clearly visible in the detection of AU26/AU27. The difference between the two is just how great the distance between facial point M and N is (see Fig 1). It is therefore no surprise that four out of five false positives of AU27 are due to activation of AU26. Another problem is the AU combination AU1 + AU4. This frequently occurring AU combination has a distinct motion pattern that cannot be simply described by adding the normal patterns of AU1 and AU4. In the training set this combination does not occur, which explains the lower scores for AU1 and AU4 when the method was tested on the Cohn-Kanade database samples picturing AU1+AU4 activation[2].

Table 3: Results of database generalization, training the classifier on data from the MMI facial expression database and testing on data from the Cohn-Kanade database. The second column lists the number of positive/negative sessions for the specified Action Unit. A session is positive for an AU if the AU is contained in that session.

| AU | truth | cl. rate | Recall | Precision |
|---|---|---|---|---|
| 1 | 55/98 | 0.87 | 0.71 | 0.91 |
| 2 | 41/112 | 0.86 | 0.83 | 0.71 |
| 4 | 47/106 | 0.87 | 0.66 | 0.89 |
| 6 | 32/121 | 0.53 | 0.53 | 0.8 |
| 9 | 25/128 | 0.94 | 0.80 | 0.83 |
| 12 | 36/117 | 0.84 | 0.92 | 0.61 |
| 25 | 94/59 | 0.90 | 0.90 | 0.93 |
| 26 | 20/133 | 0.82 | 0.52 | 0.39 |
| 27 | 25/128 | 0.97 | 1.00 | 0.83 |
| | Total: | 0.84 | 0.76 | 0.77 |

## 5 Conclusions

In this paper we present a facial point tracking scheme using particle filtering with factorized likelihoods enhanced with a novel observation model combining a rigid and a morphologic model. Using features calculated from tracked facial points as input of a PAL-SVM we are able to detect a similar amount of AUs as reported in [2, 5, 4, 7] with similar or higher recognition rates than those reported thus far.

The results presented in section 4.1 show clearly that a data representation based on tracked facial points is very well suited for the task of AU detection and, in turn, for the task of automated facial expression analysis. The fact that it is possible to detect AUs per frame, makes this data representation also suitable for analyzing the dynamics of facial expressions. Section 4.2 indicates that our system showed high generalizability when trained on one database and tested on another independent facial expression benchmark database. However, the results presented in section 4.2 also indicate that a richer training set is desirable.

Except the number of AUs and the temporal dynamics

[2]When we exclude all AU1+AU4 samples from the test set, AU1 has a classification rate of 0.93 and a recall of 0.87 while AU4 has a classification rate of 0.92 and a recall of 0.77.

handled, our method has also improved other aspects of automated AU detection compared to previously reported systems . The performance of the proposed system is invariant to occlusions like glasses an facial hair as long as these do not entirely occlude facial fiducial points (e.g. point E on the eyebrows). Due to the observation model used, the method performs well independently of changes in the illumnination intensity. Finally, our system is invariant to translation and in plane rotation of the face. It is invariant to out of plane rotation of the face as long as all facial points remain visible.

Two limitations of the current system are the manual initialisation of the facial points in the first frame of an input face video and the computational complexity of the PFFL technique. To address the first limitation we are currently investigating methods to automatically detect the facial point locations in the first frame. This improvement would be a big step toward a fully automated facial expression detection system. Computational complexity however still remains a problem to be addressed in the future.

In the near future we plan to conduct an even larger experimental study that would evaluate the generalizability of our system using larger and more comprehensive training and testing data sets. Furthermore, because the reliability of the system is rather high, we will be able to perform in-depth studies of the temporal dynamics of facial expressions, which form the main focus of our future research in the field.

## Acknowledgements

The authors would like to thank Jeffrey Cohn of the University of Pittsburgh for providing the Cohn-Kanade database. The work of M.F. Valstar and I. Patras has been supported by the Netherlands BSIK-MultimediaN-N2 Interaction project. The work of M. Pantic has been supported by the Netherlands Organization for Scientific Research (NWO) Grant EW-639.021.202.

## References

[1] J.A. Russell and J.M. Fernandez-Dols, Eds., *The Pshychology of Facial Expression*, New York: Cambridge University Press, 1997

[2] M. Pantic and L.J.M. Rothkrantz, "Toward an Affect-Sensitive Multimodal Human-computer Interaction", *Proceedings of the IEEE*, vol. 91, no. 9, pp. 1370-1390, 2003

[3] P. Ekman and W.V. Friesen, *The Facial Action Coding System: A Technique for the Measurement of Facial Movement*, San Francisco: Consulting Psychologist, 1976

[4] Y.Tian, T. Kanade and J.F. Cohn, "Recognizing action units for facial expression analysis", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 97-115, 2001

[5] M. Pantic, I. Patras and L.J.M. Rothkrantz, "Facial action recognition in face profile image sequences", *Proc. IEEE Int'l Conf. Multimedia and Expo*, vol 1, pp. 37-40, 2002

[6] M.F. Valstar, I. Patras and M. Pantic, "Motion history for facial action detection in video", *Proc. IEEE Int'l Conf. Systems Man and Cybernetics,* vol 1, pp. 635-640, 2004

[7] M.S. Bartlett, G. Littlewort,C. Lainscsek, I. Fasel, J. Movellan, "Machine Learning Methods for Fully Automatic Recognition of Facial Expressions and Actions", *Proc. IEEE Int'l Conf. Systems Man and Cybernetics,* vol 1, pp. 592-597, 2004

[8] I. Patras and M. Pantic, "Particle filtering with factorized likelihoods for tracking facial features", *Proc. IEEE Int'l Conf. Automatic Face and Gesture Recognition*, pp. 97-102, 2004

[9] P. Mitra, C.A. Murthy, S.K. Pal, "A probabilistic active learning support vector learning algorithm", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 26, no. 3, pp. 413-418, 2004

[10] M.Pantic, M.F. Valstar, R. Rademaker and L. Maat, "Web-based database for facial expression analysis", *Proc. of the IEEE Int'l Conf. Multimedia and Expo, accepted for publication*, 2005

[11] T. Kanade, J. Cohn and Y. Tian, "Comprehensive database for facial expression analysis", *Proc. IEEE Int'l Conf. Automatic Face and Gesture Recognition*, pp. 46-53, 2000

[12] M.K. Pitt and N. Shephard. Filtering via simulation: auxiliary particle filtering. *J. American Statistical Association,* vol. 94, pp. 590-599, 1999

[13] M. Pantic and L.J.M. Rothkrantz, "Facial action recognition for facial expression analysis from static face images", *IEEE trans. on Systems, Man and Cybernetics Part B*, vol. 34, pp. 1449-1461, 2004

[14] M. Isard and A. Blake, "Condensation - conditional density propagation for visual tracking", *Int'l J. Computer Vision*, pp. 5-28, 1998