Dynamic Cascades with Bidirectional Bootstrapping for Action Unit Detection in Spontaneous Facial Behavior

Yunfeng Zhu, Fernando De la Torre, Jeffrey F. Cohn, Associate Member, IEEE, and Yu-Jin Zhang, Senior Member, IEEE

Abstract—Automatic facial action unit detection from video is a long standing problem in facial expression analysis. Research has focused on registration, choice of features, and classifiers. A relatively neglected problem is the choice of training images. Nearly all previous work uses one or the other of two standard approaches. One approach assigns peak frames to the positive class and frames associated with other actions to the negative class. This approach maximizes differences between positive and negative classes, but results in a large imbalance between them, especially for infrequent AUs. The other approach reduces imbalance in class membership by including all target frames from onsets to offsets in the positive class. However, because frames near onsets and offsets often differ little from those that precede them, this approach can dramatically increase false positives. We propose a novel alternative, dynamic cascades with bidirectional bootstrapping (DCBB) to select training samples.

Using an iterative approach, DCBB optimally selects positive and negative samples in the training data. Using Cascade Adaboost as basic classifier, DCBB exploits the advantages of feature selection, efficiency, and robustness of Cascade Adaboost. To provide a real-world test, we used the M3 (also known as RU-FACS) database of nonposed behavior recorded during interviews. For all tested action units, DCBB improved AU detection relative to alternative approaches.

Index Terms—Facial expression analysis, action unit detection, FACS, dynamic cascade boosting, bidirectional bootstrapping.

1 INTRODUCTION

The face is one of the most powerful channels of nonverbal communication. Facial expression provides cues about emotional response, regulates interpersonal behavior, and communicates aspects of psychopathology. To make use of the information afforded by facial expression, Ekman and Friesen [1] proposed the Facial Action Coding System (FACS). FACS segments the visible effects of facial muscle activation into “action units (AUs)”. Each action unit is related to one or more facial muscles. These anatomic units may be combined to represent more molar facial expressions. Emotion-specified joy, for instance, is represented by the combination of AU6 (cheek raiser, which results from contraction of the orbicularis oculi muscle) and AU12 (lip-corner puller, which results from contraction of the zygomatic major muscle). The FACS taxonomy was developed by manually observing live and recorded facial behavior and to a lesser extent by recording the electrical activity of underlying facial muscles [2]. Because of its descriptive power, FACS has become the state of the art in manual measurement of facial expression [3] and is widely used in studies of spontaneous facial behavior [4]. For these and related reasons, much effort in automatic facial image analysis seeks to automatically recognize FACS action units [5], [6], [7], [8], [9].

Automatic detection of AUs from video is a challenging problem for several reasons. Non-frontal pose and moderate to large head motion make facial image registration diffi-
cult, large variability occurs in the temporal scale of facial gestures, individual differences occur in shape and appearance of facial features, many facial actions are inherently subtle, and the number of possible combinations of 40+ individual action units is in the thousands. More than 7000 action unit combinations have been observed [4]. Previous efforts have emphasized types of features and classifiers. Features have included shape and various appearance features, such as grayscale pixels, edges, and appearance (e.g., canonical appearance, Gabor, and SIFT descriptors). Classifiers have included support vector machines (SVM) [10], boosting [11], hidden markov models (HMM) [12] and dynamic Bayesian networks (DBN) [13]. Little attention has been paid to the assignment of video frames to positive and negative classes. Typically, assignment has been done in one of two ways. One is to assign to the positive class those frames that occur at the peak of each AU or proximal to it. Peaks refer to the maximum intensity of an action unit between the frame at which begins ("onset") and ends ("offset"). Negative class then is chosen by randomly sampling other AUs, including AU 0 or neutral. This approach suffers at least two drawbacks: (1) the number of training examples will often be small, which results in a large imbalance between positive and negative frames; and (2) peak frames may provide too little variability to achieve good generalization. These problems may be circumvented by following an alternative approach; that is to include all frames from onset to offset in the positive class. This approach improves the ratio of positive to negative frames and increases representativeness of positive examples. The downside is confusability of positive and negative classes. Onset and offset frames and many of those proximal or even further from them may be indistinguishable from the negative class. As a consequence, the number of false positives may dramatically increase.

To address these issues, we propose an extension of AdaBoost [14], [15], [16] called Dynamic Cascades with Bidirectional Bootstrapping (DCBB). Fig. 1 illustrates the main idea. Having manually annotated FACS data with onset, peak and offset, the question we address is how best to select the AU frames for the positive and negative class.

In contrast to previous approaches to class assignment, DCBB automatically distinguishes between strong, subtle and ambiguous AU frames for AU events of different intensity. Strong frames correspond to the peaks and the ones proximal to them; ambiguous AU frames are proximal to onsets and offsets; subtle AU frames occur between strong and ambiguous. Strong and subtle frames are assigned to the positive class. By distinguishing between these three types, DCBB maximizes the number of positive frames while reducing confusability between positive and negative classes.

For high intensity AUs in comparison with low intensity AUs, the algorithm will select more frames for the positive class. Some of these frames may be similar in intensity to low intensity AUs. Similarly, if multiple peaks occur between an onset and offset, DCBB assigns multiple segments to the positive class. See Fig. 1 for an example. Strong and subtle but not ambiguous AU frames are assigned to the positive class. For the negative class, DCBB proposes a mechanism, which is similar as Cascade AdaBoost to optimize that as well, the principles are that the weight of misclassified negative class will be increased during training step of each weak classifier, and don’t learning to much at each cascade stage. Moreover, the positive class is changed at each iteration, while the corresponding negative class is reselected again.

In experiments, we evaluated the validity of our approach to class assignment and selection of features. In the first experiment, we illustrate the importance of selecting the right positive samples for action unit detection. In the second we compare DCBB with standard approaches based on SVM and AdaBoost.

The rest of the paper is organized as follows. Section II reviews previous work on automatic methods for action unit detection. Section III describes pre-processing steps for alignment and feature extraction. Section IV gives details of our proposed DCBB method. Section V provides experimental results in non-posed, naturalistic video. For experimental evaluation, we used FACS-coded interviews from the M3 (also known as RU-FACS) database [17], [18]. For all action units tested, DCBB outperformed alternative approaches.

2 PREVIOUS WORK

This section describes previous work on FACS and on automatic detection of AUs from video.

2.1 FACS

The Facial Action Coding System (FACS) [1] is a comprehensive, anatomically-based system for measuring nearly all visually discernible facial movement. FACS describes facial activity on the basis of 44 unique action units (AUs), as well as several categories of head and eye positions and movements. Facial movement is thus described in terms of constituent components, or AUs. Any facial expression may be represented as a single AU or a combination of AUs. For example, the felt, or Duchenne, smile is indicated by movement of the zygomatic major (AU12) and orbicularis oculi, pars lateralis (AU6). FACS is recognized as the most comprehensive and objective means for measuring facial movement currently available, and it has become the standard for facial measurement in behavioral research in psychology and related fields. FACS coding procedures allow for coding of the intensity of each facial action on a 5-point intensity scale (which provides a metric for the degree of muscular contraction) and for measurement of the timing of facial actions. FACS scoring produces a list of AU-based descriptions of each facial event in a video record. Fig. 2 shows an example for AU12.

2.2 Automatic FACS detection from video

Two main streams in the current research on automatic analysis of facial expressions consider emotion-specified
expressions (e.g., happy or sad) and anatomically based facial actions (e.g., FACS). The pioneering work of Black and Yacoob [19] recognizes facial expressions by fitting local parametric motion models to regions of the face and then feeding the resulting parameters to a nearest neighbor classifier for expression recognition. De la Torre et al. [20] used condensation and appearance models to simultaneously track and recognize facial expression. Chang et al. [21] learnt a low dimensional Lipschitz embedding to build a manifold of shape variation across several people and then use I-condensation to simultaneously track and recognize expressions. Lee and Elgammal [22] employed multi-linear models to construct a non-linear manifold that factorizes identity from expression.

Several promising prototype systems were reported that can recognize deliberately produced AUs in either near frontal view face images (Bartlett et al., [23]; Tian et al., [8]; Pantic & Rothkrantz, [24]) or profile view face images (Pantic & Patras, [25]). Although high scores have been achieved on posed facial action behavior [13], [26], [27], accuracy tends to be lower in the few studies that have tested classifiers on non-posed facial behavior [28], [11], [29]. In non-posed facial behavior, non-frontal views and rigid head motion are common, and action units are often less intense, have different timing, and occur in complex combinations [30]. These factors have been found to reduce AU detection accuracy [31]. Non-posed facial behavior is more representative of facial actions that occur in real life, which is our focus in the current paper.

Most work in automatic analysis of facial expressions differs in the choice of facial features, representations, and classifiers. Barlett et al. [11], [23], [18] used SVM and AdaBoost in texture-based image representations to recognize 20 action units in near-frontal posed and non-posed facial behavior. Valstar and Pantic [32], [25], [33] proposed a system that enables fully automated robust facial expression recognition and temporal segmentation of onset, peak and offset from video of mostly frontal faces. They used particle filtering to track facial features, Gabor-based representations, and a combination of SVM and AdaBoost in texture-based image representations to achieve 75% accuracy on action units.

To the best of our knowledge, no previous work has considered strategies for selecting training samples or evaluated their importance in AU detection. This is the first paper to propose an approach to optimize the selection of positive and negative training samples. We show that our approach consistently improves action unit detection relative to previous approaches.

### 3 Facial feature tracking and image features

This section describes the system for facial feature tracking using active appearance models (AAMs), and extraction and representation of shape and appearance features for input to the classifiers.

#### 3.1 Facial tracking and alignment

Over the last decade, appearance models have become increasingly important in computer vision and graphics. Parameterized Appearance Models (PAMs) (e.g. Active Appearance Models [36], [37], [38] and, Morphable Models [39]) have been proven useful for detection, facial feature alignment, and face synthesis. In particular, Active Appearance Models (AAMs) have proven an excellent tool for aligning facial features with respect to a shape and appearance model. In our case, the AAM is composed of 60 landmarks that deform to fit perturbations in facial features. Person-specific models were trained on approximately 5% of the video [38]. Fig. 3 shows an example of AAM tracking of facial features in a single subject from the M3 (also known as RU-FACS) [17], [18] video data-set.

After tracking facial features using AAM, a similarity transform registers facial features with respect to an average face (see middle column in Fig. 4). The face is normalized to have 212 x 212 pixels in all our experiments. To extract...
appearance representations in areas that have not been explicitly tracked (e.g. nasolabial furrow), we use a backward piece-wise affine warp with Delaunay triangulation to set up the correspondence. Fig. 4 shows the two step process for registering the face to a canonical pose for AU detection. Purple squares represent tracked points and blue dots represent non-tracked meaningful points. The dashed blue line shows the mapping between the point in the mean shape and the corresponding points on the original image. This two-step registration proved particularly important to detect low intensity action units.

3.2 Appearance features

Appearance features for AU detection [11], [40] outperformed shape only features for some action units, see Lucey et al. [34], [41], [42] for a comparison. In this section, we explore the use of the SIFT [43] descriptors to be used as appearance features.

Given feature points tracked with AAMs, SIFT descriptors are first computed around the points of interest. SIFT descriptors are computed from the gradient vector for each pixel in the neighborhood to build a normalized histogram of gradient directions. For each pixel within a subregion, SIFT descriptors add the pixel’s gradient vector to a histogram of gradient directions by quantizing each orientation to one of 8 directions and weighting the contribution of each vector by its magnitude.

4 Dynamic Cascades with Bidirectional Bootstrapping (DCBB)

This section explores the use of a dynamic boosting techniques to select the positive and negative samples that improve detection performance in AU detection.

Bootstrapping [44] is a resampling method that is compatible with many learning algorithms. During the bootstrapping process, the active sets of negative examples are extended by examples that were misclassified by the current classifier. In this section, we propose Bidirectional Bootstrapping, a method to bootstrap both positive and negative samples.

Bidirectional Bootstrapping beings by selecting as positive samples only the peak frames and uses Classification and Regression Tree (CART) [45] as a weak classifier (Initial learning in Section 4.1). After this initial training step, Bidirectional Bootstrapping extends the positive samples from the peak frames to proximal frames and redefines new provisional positive and negative training sets (Dynamic learning in Section 4.2). The positive set is extended by including samples that are classified correctly by the previous strong classifier(Cascade AdaBoost in our algorithm), the negative set is extended by examples misclassified by the same strong classifier, thus emphasizing negative samples close to the decision boundary. With the bootstrapping of positive samples, the generalization ability of the classifier is gradually enhanced. The active positive and negative sets then are used as an input to the CART that returns a hypothesis, which updates the weights in the manner of Gentle AdaBoost [46], and the training continues until the variation between previous and current Cascade AdaBoost become smaller than a defined threshold. Fig. 5 illustrates the process. In Fig. 5 P is potential positive data set, Q is negative data set(Negative Pool), P0 is the positive set in Initial learning step, Pw is the active positive set in each iteration, the size of solid circle illustrate the intensity of AU samples, the right ellipses illustrate the spreading of dynamic positive set. See details in Algorithm 1 and 2.

4.1 Initial training step

This section explains the initial training step for DCBB. In the initial training step we select the peaks and the two neighboring samples as positive samples, and a random sample of other AUs and non-AUs as negative samples. As in standard AdaBoost [14], we define the false positive target ratio (Ft), the maximum acceptable false positive ratio per cascade stage (fT), and the minimum acceptable true positive ratio per cascade stage (dT). The cascade classifier we used CART. The initial training step applies standard AdaBoost using CART as a weak classifier as summarized in Algorithm 1.
Input:
- Positive data set $P_0$ (contains AU peak frames $p$ and $p \pm 1$);
- Negative data set $Q$ (contains other AUs and non-AUs);
- Target false positive ratio $F_r$;
- Maximum acceptable false positive ratio per cascade stage $f_r$;
- Minimum acceptable true positive ratio per cascade stage $d_r$;

Initialize:
- Current cascade stage number $t = 0$;
- Current overall cascade classifier’s true positive ratio $D_{t} = 1.0$;
- Current overall cascade classifier’s false positive ratio $F_{t} = 1.0$;
- $S_0 = \{P_0, Q_0\}$ is the initial working set, $Q_0 \subset Q$.

While $F_t > F_r$
1) $t = t + 1; f_t = 1.0$; Normalize the weights $\omega_{k,i}$ for each sample $x_i$ to guarantee that $\omega_t = \{\omega_{t,i}\}$ is a distribution.
2) While $f_t > f_r$
   a) For each feature $\phi_m$, train a weak classifier on $S_0$ and find the best feature $\phi_t$ (the one with the minimum classification error).
   b) Add the feature $\phi_t$ into the strong classifier $H_t$, update the weight in Gentle AdaBoost manner.
   c) Evaluate on $S_0$ with the current strong classifier $H_t$, adjust the rejection threshold under the constraint that the true positive ratio does not drop below $d_r$.
   d) Decrease threshold until $d_r$ is reached.
   e) Compute $f_t$ under this threshold.

END While
3) $F_{t+1} = F_t \times f_t; D_{t+1} = D_t \times d_r$; keep in $Q_0$ the negative samples that the current strong classifier $H_t$ misclassified (current false positive samples), record its size as $K_{f_t}$.
4) Repeat using detector $H_t$ to bootstrap false positive samples from negative $Q$ randomly until the negative working set has $N_q$ samples.

END While

Output:
A t-levels cascade where each level has a strong boosted classifier with a set of rejection thresholds for each weak classifier. The final training accuracy figures are $F_t$ and $D_t$.

Algorithm 1: Initial learning

4.2 Dynamic learning

Once a cascade of peak frame detectors is learned in the initial learning stage (Section 4.1), we are able to enlarge the positive set to increase the discriminative performance of the whole classifier. The AU frames detector will become stronger as new AU positive samples are added during the training steps, and the distribution of positive and negative samples will become more representative of the whole training data. A constraint scheme is applied in dynamic learning to avoid adding ambiguous AU frames to the dynamic positive set. The algorithm is summarized in Algorithm 2.

Input
- cascade detector $H_0$, from the Initial Learning step;
- Dynamic working set $S_D = \{P_D, Q_D\}$;
- All the frames in this action unit as potential positive samples $P = \{P_p, P_n\}$. $P_s$ contains the strong positive samples, $P_0$ contains peak related samples described above, $P_0 \subset P_s$. $P_c$ contains obscure positive samples;
- A large negative data set $Q$, which contains all the other AUs and non-AUs. Its size is $N_a$.

Update positive working set by spreading in $P$ and update negative working set by bootstrap in $Q$ dynamic cascade learning:

Initialize: We set the value of $N_p$ as the size of $P_0$. The size of the old positive data set is $N_{p,old} = 0$. Current diffusing stage is $t = 1$.

While $(N_p - N_{p,old})/N_p > 0.1$
1) **AU Positive Spreading:** $N_{p,old} = N_p$. Using current detector on the potential positive data set $P$ to pick up more positive samples, $P_{sp}$ are all the positive samples that determined by the cascade classifier $H_{t-1}$.
2) **Constrain the spreading:** $k$ is the index of current AU event, $i$ is the index of current frame in this event. Calculate the similarity values (Eq. 1) between the peak frame in event $k$ and all peak frames with the lowest intensity value ‘A’, the average similarity value is $S_k$. Calculate the similarity value between frame $i$ and peak frame in event $k$, its value is $S_{ki}$, if $S_{ki} < 0.5 \times S_k$, frame $i$ will exclude from $P_{sp}$.
3) After the above step, the remaining positive work set is $P_w = P_{sp}, N_w = \text{size of } P_{sp}$. Using $H_{t-1}$ detector to bootstrap false positive samples from the negative set $Q$ until the negative working set $Q_w$ has $N_q = \beta \times R_0 \times N_a$ samples, $N_a$ is different in AUs.
4) Train the cascade classifier $H_t$ with the dynamic working set $\{P_w, Q_w\}$. As $R_t$ varies, the maximum acceptable false positive ratio per cascade stage $f_m$ also becomes smaller (Eq. 2).
5) $t = t + 1$; empty $P_w$ and $Q_w$.

END While

Algorithm 2: Dynamic learning

In eq.1, $n$ is the total number of AU sections with intensity ‘A’, and $m$ is the length of the AU features. The similarity description used in eq.1 is the Radial Basis Function between the appearance representation of two
frames.

\[ S_k = \frac{1}{n} \sum_{j=1}^{n} \text{Sim}(f_k, f_j), \quad j \in [1:n] \]

\[ S_{ik} = \text{Sim}(f_i, f_k) = e^{-\frac{\text{Dist}(i,k)/\max(\text{Dist}(:,k))}{2}} \]

\[ \text{Dist}(i, k) = \left[ \sum_{j=1}^{m} (f_{kj} - f_{ij})^2 \right]^{1/2}, j \in [1:m] \quad (1) \]

The dynamic positive work set becomes larger but the negative samples pool is finite, so \( f_m \), need to be changed dynamically, and \( N_q \) is decided by \( N_a \) as different AU has different size of the negative samples pool. Some AUs (e.g., AU12) are likely to be occur more often than others. Instead of tuning these thresholds one by one, we assume that the false positive ratio \( f_m \) changes exponentially in each stage \( t \), which means

\[ f_m = f_r \times (1 - e^{-\alpha R_t}) \quad (2) \]

In our experiment, we set \( \alpha = 0.2 \) and \( \beta = 0.04 \) respectively because those values are suitable for all the AUs to avoid lacking of useful negative samples in M3 database. After the spreading stage, the ratio between positive and negative samples becomes balanced, except for some rare AUs (e.g., AU4, AU10) which keep unbalanced because of the scarceness of positive frames in the database.

5 Experiments

DCBB iteratively samples training images and then uses Cascade AdaBoost for classification. We evaluated both, as well as the use of additional appearance features. To evaluate the efficacy of iteratively sampling training images, Experiment 1 compared DCBB with the two standard approaches. They are selecting only peaks and alternatively selecting all frames between onsets and offsets. Experiment 1 thus evaluated whether iteratively sampling training images increased AU detection. Experiment 2 evaluated the efficacy of Cascade AdaBoost relative to SVM when iteratively sampling training images. In our implementation, DCBB uses Cascade AdaBoost, but other classifiers might be used instead. Experiment 2 informs whether better results might be achieved using SVM. Experiment 3 explored the use of three appearance descriptions (Gabor, SIFT and DAISY) in conjunction with DCBB (Cascade AdaBoost as classifier).

5.1 Database

The three experiments all used the M3 (also known as RU-FACS) database [17], [18]. M3 consists of video-recorded interviews of 34 men and women of varying ethnicity. Interviews were approximately two minutes in duration. Video from five subjects could not be processed for technical reasons (e.g., noisy video), which resulted in usable data from 29 participants. Meta-data included manual FACS codes for AU onsets, peaks and offsets. Because some AUs occurred too infrequently, we selected the 13 AUs that occur more than 20 times in the database (i.e. 20 or more peaks). These AUs are: AU1, AU2, AU4, AU5, AU6, AU7, AU10, AU12, AU14, AU15, AU17, AU18, and AU23. Fig. 6 shows the number of frames that each AU occurred and their average duration. Fig. 5 illustrates a representative time series for several AUs from subject S010. Blue asterisks represent onsets, red circles peak frames, and green plus signs the offset frames. In each experiment, we randomly selected 19 subjects for training and the other 10 subjects for testing.

5.2 Experiment 1: Iteratively sampling training images with DCBB

This experiment illustrates the effect of iteratively selecting positive and negative samples (i.e. frames) while training the Cascade AdaBoost. As a sample selection mechanism on top of the Cascade AdaBoost, DCBB could be applied to other classifiers as well. The experiment investigates that whether the iteratively selecting training samples strategy is better than the strategy using only peak AU frames and the strategy using all the AU frames as positive samples. For different positive samples assignments, the negative samples is defined by the method used in Cascade AdaBoost.

We apply the DCBB method, described in Section IV and use appearance features based on SIFT descriptors (Section III). For all AUs the SIFT descriptors are built using a square of 48 \times 48 pixels for twenty feature points for the lower face AUs or sixteen feature points for upper face (see Fig. 4). We trained 13 dynamic cascade classifiers, one for each AU, as described in Section IV, using a one versus all scheme for each AU.

Fig. 6. Top) Frame number from onset to offset for the 13 most frequent AUs in the M3 dataset. Bottom) The average duration of AU in frames.
Fig. 7. AUs characteristics in subject S010, duration, intensity, onset, offset, peak. (frames as unit in X axis, Y axis is the intensity of AUs)

Fig. 8. The spreading of positive samples during each dynamic training step for AU12. See text for the explanation of the graphics.

in the first step are represented by green asterisks, in the second iteration by red crosses, in the third iteration by blue crosses, and in the final iteration by black circles. Observe that in the case of high peak intensity, subfigures 3 and 8 (top right number in the similarity plots), the final selected positive samples contain areas of low similarity values. When AU intensity is low, subfigure 7, positive samples are selected if they have a high similarity with the peak, which reduces to the number of false positives. The ellipses and rectangles in the figures contain frames that are selected as positive samples, and correspond to strong and subtle AUs in Fig. 1. The triangles correspond to frames between the onset and offset that are not selected as positive samples, and represent ambiguous AUs in Fig. 1.

*************** change ***************

Table 1 show the number of frames at each level of intensity and the percentage of each intensity that were selected as positive by DCBB in training dataset. The 'A-E' in left column refers to the level of AU intensities, The 'Num' and 'Pct' in the top rows refers to the number of AU frames at each level of intensity and the percentage of each intensity that were selected as positive examples at the last iteration of dynamic learning step respectively.
If there is no AU frames in one level of intensity, the 'Pct' value will be 'NaN'. From this table, we can find that the mainly AU frames that were selected by DCBB concentrate in the level 'C', 'D' and 'E', moreover, most of the AU frames in level 'A' were ignored by DCBB during the sampling of positive examples.

Fig. 9 shows the Receiver-Operator Characteristic (ROC) curve for testing data (subjects not in the training) using DCBB in the 13 AUs. The ROC is obtained by plotting true positives ratios against false positives ratios for different decision threshold values of the classifier. In each figure (representing one AU) there are five or six ROCs: initial learning corresponds to running on only the peaks (which is same as Cascade Adaboost without the DCBB strategy); spread x corresponds to running DCBB x times; All denotes using all frames between onset and offset. The first number between lines | in Fig. 9 denotes the area under the ROC; the second number is the size of positive samples in the testing dataset; and separated by / is the number of negative samples in the testing dataset. The third number denotes the size of positive samples in training working sets and separated by / the total frames of target AU in training data sets. AU5 has the minimum number of training examples and AU12 has the largest number of examples. We can observed that the area under the ROC for frame-by-frame detection is improved gradually during each learning stage and the performance improves faster for AU4, AU5, AU10, AU14, AU15, AU17, AU18, AU23 than for AU1, AU2, AU6 and AU12 during Dynamic learning. It is also important to notice that using all the frames between the onset and offset ('All') typically degrades the detection performance. The table of Fig. 9 shows the area under the ROC using only the peak, all frames or the DCBB strategy. Clearly the DCBB outperforms standard approaches that use the peak or all frames [40], [18], [34], [10], [29].

There are two parameters in the DCBB algorithms that are manually tuned and remain the same in all experiments. One parameters specifies the minimum similarity value below which no positive sample will be selected. This threshold is based on the similarity equation (eq. 1). It was set at 0.5 after preliminary testing found results stable within a range of 0.2 to 0.6. The second parameter is the stopping criterion. We consider that the algorithm has converged if the number of new positive samples between iterations is smaller than 10%. We ran experiments for different values ranging from 5% to 15%, values lower than 15% did not change the performance in the ROC curves. In the extreme case of being 0%, the algorithm will not stop but the ROC will not improve. Typically the algorithm converges within three or four iterations.

### 5.3 Experiment 2: Comparing Cascade Adaboost and Support Vector Machine (SVM) when used with iteratively sampled training frames

SVM and AdaBoost are commonly used classifiers for AU detection. In this experiment, we compared AU detection by DCBB (Cascade AdaBoost as classifier) and SVM using shape and appearance features. We found that for both types of features, DCBB achieved more accurate AU detection.

For compatibility with the previous experiment, data from the same 19 subjects as above were used for training and the other 10 for testing. Results are reported for the 13 most frequently observed AU. Other AU occurred too infrequently (i.e. fewer than 20 occurrences) to obtain reliable results and thus were omitted. Classifiers for each AU were trained using a one versus-all strategy. The ROC curves for the 13 AUs are shown in Fig. 10. For each AU, six curves are shown, one for each combination of training features and classifiers. 'App+DCBB' refers to DCBB using appearance features; 'Peak+Shp+SVM' refers to SVM using shape features trained on the peak frames (and two adjacent frames) [10]; 'Peak+App+SVM' [10] refers to train a SVM using appearance features trained on the peak frame (and two adjacent frames); 'All+Shp+SVM' refers to SVM using shape features trained on all frames between onset and offset; 'All+PCA+App+SVM' refers to SVM using appearance features (after PCA processing) trained on all frames between onset and offset, here, in
Fig. 9. The ROCs improve with the spreading of positive samples: See text for the explanation of Peak, spread x and All.
Fig. 10. ROC curve for 13 AUs using six different methods: AU peak frames with shape features and SVM (Peak+Shp+SVM), All frames between onset and offset with shape features and SVM (All+Shp+SVM), AU peak frames with appearance features and SVM (Peak+App+SVM), Sampling 1 frame in every 4 frames between onset and offset with PCA reduced appearance features and SVM (All+PCA+App+SVM), AU peak frames with appearance features and Cascade AdaBoost(Peak+App+Cascade Boost), DCBB with appearance features (DCBB).
order to computationally scale in memory space, we reduced the dimensionality of the appearance features using principal component analysis (PCA) that preserves 98% of the energy. 'Peak+App+Cascade Boost' refers to the use of the peak frame with appearance features and Cascade AdaBoost [14] classifier (will be equivalent to the first step in DCBB). As can be observed in the figure, DCBB outperformed SVM in all AUs (except AU18) using either shape or appearance when training in the peak (and two adjacent frames). When training the SVM with shape and using all samples the DCBB outperformed in eleven out of thirteen AUs. In the SVM training, the negative samples were selected randomly (but the same negative samples when using either shape or appearance features). The ratio between positive and negative samples was fixed to 30. Compared with the Cascade AdaBoost (first step in DCBB that only uses the peak and two neighbor samples), DCBB improved the performance in all AUs.

Interestingly, the performance for AU4, AU5, AU14, AU18 using the method 'All+PCA+App+SVM' was better than 'DCBB'. 'All+PCA+App+SVM' uses appearance features and all samples between onset and offset. Similarly, we sampled one frame every four frames between onset and offset. All parameters in SVM were selected using cross-validation. It is interesting to observe that AU4, AU5, AU14, AU15 and AU18 are the AUs that have very few total training samples (only 1749 total frames for AU4, 797 frames for AU5, 4089 frames for AU14, 2134 frames for AU15, 1122 frames for AU18), and when having very little training data the classifier can benefit from using all samples. An other result should be mentioned is that all the winners are using appearance feature instead of shape feature, that is because the head pose variation in M3 dataset may impair shape features more than SIFT descriptors. Using unoptimized MATLAB code, training DCBB typically requires one to three hours depending on the AU and number of training samples.

5.4 Experiment 3: Appearance descriptors for DCBB

Appearance features have proven more robust representation than shape features for most AUs. In addition to presenting and evaluating DCBB, we explore the representational power of two edge-based appearance descriptors (SIFT and DAISY descriptors) for AU detection, and compare it to the traditional Gabor representation [40], [23].

Similar in spirit to SIFT descriptors, DAISY descriptors are an efficient feature descriptor based on histograms that have been used to match stereo images [47]. DAISY descriptors use circular grids instead of SIFT descriptors’ regular grids; the former have been found to have better localization properties [48] and to outperform many state-of-the-art feature descriptors for sparse point matching [49]. At each pixel, DAISY builds a vector made of values from the convolved orientation maps located on concentric circles centered on the location. The amount of Gaussian smoothing is proportional to the radius of the circles.

In the following experiment, we compare the performance on AU detection for three appearance descriptors Gabor, SIFT and DAISY while using DCBB with Cascade Adaboost. We used the same 19 subjects for training and 10 for testing as the previous experiments. Fig. 11 shows the ROC detection curves for the lower AUs using Gabor filters at eight different orientations and five different scales, DAISY [49] and SIFT [43]. In most of AUs, SIFT and DAISY descriptors have shown similar performance, being DAISY faster to compute than the SIFT descriptors. For a fair comparison Gabor, DAISY and SIFT descriptors were computed in the same locations in the image (twenty feature points in lower face, see Fig. 12), and using the same training and testing data.

![Fig. 12. Using different number of feature points](image)

An important parameter largely conditioning the performance of appearance-based descriptors is the number and location of features selected to build the representation. For computational considerations it is not practical to build the appearance representation in all possible regions of interest. In the following experiment, we test the robustness in performance varying the location where the appearance representation is built (i.e. (2, 12, 18, 20) points) in the mouth area. In Fig. 12 the big red circles represent the selection of two feature descriptors, adding blue squares and small red circles the descriptor has 12 features. Including the purple triangle and small black square the descriptor would contain 18 and 20 points respectively. The performance of the ROCs is consistent with our intuition, the more features we add the better is the performance. This behavior does not necessarily occur with all classifiers, but boosting provides a feature selection mechanism that avoids overfitting even for large feature spaces.

6 Conclusions

An unexplored problem and critical to the success of automatic action unit detection is the selection of the positive and negative samples. This paper proposes dynamic cascade bidirectional bootstrapping (DCBB) to best select positive and negative training samples. With few exceptions, by selecting the optimal positive samples, DCBB achieved better detection performance than the standard approach of selecting either peak frames or all frames between the onsets and offsets. This results was found using SVMs.
and AdaBoost. We also compared three common used appearance features and compared different number of sampling points, the results show than the SIFT and DAISY features are more efficient than Gabor under same number of sampling points, while more sampling points can get better performance. Although promising results have been shown, there are several issues that remain unsolved. By using DCBB, we got a strong Cascade Adaboost Classifier finally, which has optimized features and training samples by bidirectional bootstrapping, however, it does not mean there is no use of the Cascade Adaboost Classifiers which trained during each iterations of dynamic learning. Actually, for each AU, we can got 3—5 Cascade Adaboost Classifiers during dynamic learning, when they were used for AU detection one by one, they will have overlapping area on the label’s sequences of the AU detecting results, these will reflect the dynamic patterns of AU events. In future work, We plan to model the dynamic patterns around the onset and offset of AU events and exclude false positive samples for all AUs. we also plan to explore how to extend this
approach to other classifiers such as Gaussian Processes or Support Vector Machines. Moreover, we plan to explore the use of these techniques in other computer vision problems such activity recognition, where the selection of the positive and negative samples might play an important role in the results.

Acknowledgments

The work was performed when the first author was at Carnegie Mellon University. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Naval Research Laboratory or the National Institutes of Health. Thanks to Tomas Simon, Feng Zhou, Zengyin Zhang for helpful comments.

References


[42] P. Lucey, J. F. Cohn, S. Lucey, S. Sridharan, and K. M. Prkachin, “Automatically detecting pain using facial actions.” in Interna-
Yunfeng Zhu received the B.Sc. degree in Information Engineering, the M.Sc. degree in Pattern Recognition and Intelligent System from Xi’an Jiaotong University in 2003 and 2006 respectively. He is currently a Ph.D. candidate at Image and Graphics Institute, in the Department of Electronic Engineering at Tsinghua University, Beijing, China. In 2008 and 2009, he became visiting student in the Robotics Institute at Carnegie Mellon University (CMU), Pittsburgh, PA. His research interests include computer vision, machine learning, facial expression recognition and 3D facial model reconstruction.

Fernando De la Torre received the B.Sc. degree in telecommunications and the M.Sc. and Ph.D. degrees in electronic engineering from Enginyeria La Salle in Universitat Ramon Llull, Barcelona, Spain, in 1994, 1996, and 2002, respectively. In 1997 and 2000, he became an Assistant and an Associate Professor, respectively, in the Department of Communications and Signal Theory at the Enginyeria La Salle. Since 2005, he has been a Research Faculty at the Robotics Institute, Carnegie Mellon University (CMU), Pittsburgh, PA. His research interests include machine learning, signal processing, and computer vision, with a focus on understanding human behavior from multimodal sensors. He is directing the Human Sensing Lab (http://humansensing.cs.cmu.edu) and the Component Analysis Lab at CMU (http://ca.cs.cmu.edu). He has coorganized the first workshop on component analysis methods for modeling, classification, and clustering problems in computer vision in conjunction with CVPR-07, and the workshop on human sensing from video in conjunction with CVPR-06. He has also given several tutorials at international conferences on the use and extensions of component analysis methods.

Jeffrey F. Cohn earned the PhD degree in clinical psychology from the University of Massachusetts at Amherst. He is a professor of psychology and psychiatry at the University of Pittsburgh and an Adjunct Faculty member at the Robotics Institute, Carnegie Mellon University. For the past 20 years, he has conducted investigations in the theory and science of emotion, depression, and nonverbal communication. He has co-led interdisciplinary and interinstitutional efforts to develop advanced methods of automated analysis of facial expression and prosody and applied these tools to research in human emotion disorders, communication, biomedicine, biometrics, and human-computer interaction. He has published more than 120 papers on these topics. His research has been supported by grants from the US National Institutes of Mental Health, the US National Institute of Child Health and Human Development, the US National Science Foundation, the US Naval Research Laboratory, and the US Defense Advanced Research Projects Agency. He is a member of the IEEE and the IEEE Computer Society.

Yu-Jin Zhang received the Ph.D. degree in Applied Science from the State University of Lige, Lige, Belgium, in 1989. From 1989 to 1993, he was post-doc fellow and research fellow with the Department of Applied Physics and Department of Electrical Engineering at the Delft University of Technology, Delft, the Netherlands. In 1993, he joined the Department of Electronic Engineering at Tsinghua University, Beijing, China, where he is a professor of Image Engineering since 1997. In 2003, he spent his sabbatical year as visiting professor in the School of Electrical and Electronic Engineering at Nanyang Technological University (NTU), Singapore. His research interests are mainly in the area of Image Engineering that includes image processing, image analysis and image understanding, as well as their applications. His current interests are on object segmentation from images and video, segmentation evaluation and comparison, moving object detection and tracking, face recognition, facial expression detection/classification, content-based image and video retrieval, information fusion for high-level image understanding, etc. He is vice president of China Society of Image and Graphics and director of academic committee of the Society. He is also a Fellow of SPIE.