Action Unit Detection with Segment-based SVMs

Tomas Simon*, Minh Hoai Nguyen*, Fernando De La Torre, Jeffrey F. Cohn
The Robotics Institute, Carnegie Mellon University
{tsimon, minhhoai}@cmu.edu, {ftorre, jeffcohn}@cs.cmu.edu

Abstract

Automatic facial action unit (AU) detection from video is a long-standing problem in computer vision. Two main approaches have been pursued: (1) static modeling—typically posed as a discriminative classification problem in which each video frame is evaluated independently; (2) temporal modeling—frames are segmented into sequences and typically modeled with a variant of dynamic Bayesian networks. We propose a segment-based approach, kSeg-SVM, that incorporates benefits of both approaches and avoids their limitations. kSeg-SVM is a temporal extension of the spatial bag-of-words. kSeg-SVM is trained within a structured output SVM framework that formulates AU detection as a problem of detecting temporal events in a time series of visual features. Each segment is modeled by a variant of the BoW representation with soft assignment of the words based on similarity. Our framework has several benefits for AU detection: (1) both dependencies between features and the length of action units are modeled; (2) all possible segments of the video may be used for training; and (3) no assumptions are required about the underlying structure of the action unit events (e.g., i.i.d.). Our algorithm finds the best k-or-fewer segments that maximize the SVM score. Experimental results suggest that the proposed method outperforms state-of-the-art static methods for AU detection.

1. Introduction

The face is a powerful channel 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 [8] proposed the Facial Action Coding System (FACS). FACS is a comprehensive, anatomically-based system for measuring all visually discernible facial movement. It segments all facial activity on the basis of 44 unique "action units" (AUs), as well as several categories of head and eye positions and movements. Any facial event (e.g., a gesture, expression or speech component) may be a single AU or a combination of AUs. For example, the felt, or Duchenne smile, is indicated by movement of the zygomatic major (AU12, e.g., Fig. 2) and orbicularis oculi, pars lateralis (AU6). Because of its descriptive power, FACS has become the state of the art in manual measurement of facial expression and is widely used in studies of spontaneous facial behavior. Much effort in automatic facial image analysis seeks to automatically recognize FACS action units [12, 20, 25, 27].

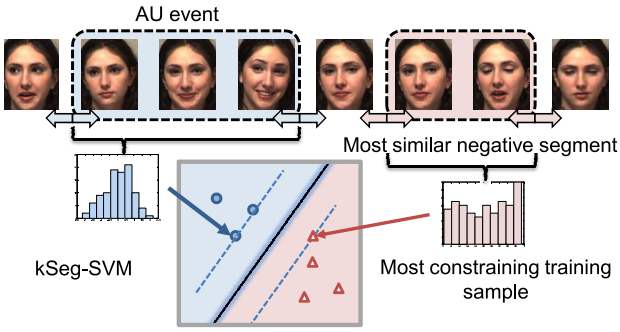


Figure 1. During testing, the AU events are found by efficiently searching over the segments (position and length) that maximize the SVM score. During training, the algorithm searches over all possible negative segments to identify those hardest to classify, which improves classification of subtle AUs.

AU detection from video is a challenging computer vision and pattern recognition problem. Some of the most important challenges are to: (1) accommodate large variability of action units across subjects; (2) train classifiers when relatively few examples for each AU are present; (3) recognize subtle AUs; (4) and model the temporal dynamics of AUs, which can be highly variable.

To address some of these issues, various approaches have been proposed. Static approaches [3, 12, 15, 25] pose AU detection as a binary- or multi-class classification problem using different features (e.g., appearance, shape) and classifiers (e.g., Boosting, SVM). The classifiers are typically trained on a frame-by-frame basis. For a given AU, the positive class comprises a subset of frames between its

*T. Simon and M. H. Nguyen contributed equally to this work.

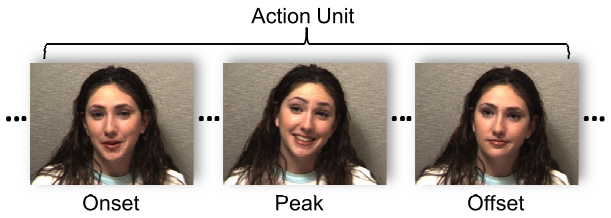


Figure 2. Left to right, evolution of an AU12 (involved in smiling), from onset, peak, to offset.

onset and offset, and the negative class comprises a subset of frames labeled as neutral or other AUs. Temporal models, such as modifications of dynamic Bayesian networks [6, 10, 21, 27, 29] model the dynamics of the AU as transitions in a partially observed state space.

Although static and dynamic approaches have achieved high performance on most posed facial expression databases [3, 23, 25], accuracy tends to be much lower in studies that test on non-posed facial expressions [3, 12, 31]. Non-posed expressions are challenging. They are less stereotypic, more subtle, more likely to co-occur with other AUs, and more-often confounded by increased noise due to variation in pose, out-of-plane head motion, and co-occurring speech. They also may be more complex temporally. Segmentation into onset, one or more local peaks, and offset must be discovered.

For non-posed facial behavior, static approaches may be more susceptible to noise because independent decisions are made on each frame. Similarly, hidden state temporal models suffer the drawbacks of needing either an explicit definition of the latent state of all frames, or the need to simultaneously learn a state sequence and state transition model that fits the data, resulting in a high-dimensional minimization problem with typically many local minima. In this paper, we propose a segment-based SVM, kSeg-SVM, a temporal extension of the spatial Bag-of-Words (BoW) [5] approach trained with a Structured Output SVM framework (SO-SVM) [28], which results in a convex optimization problem.

kSeg-SVM is inspired by the success of the spatial BoW sliding-window model [5] that has been used in difficult object detection problems. We pose the AU detection problem as one of detecting temporal events (segments) in time series of visual features. Events correspond to AUs, including all frames between onset and offset (see Fig. 1). kSeg-SVM represents each segment as a BoW; however, the standard histogram of entries is augmented with a soft-clustering assignment of words to account for smoothly-varying signals. Given several videos with AU labeled events, kSeg-SVM learns the SVM parameters that maximize the response on positive segments (AU to be detected) and minimize the response in the rest of the segments (all other positions and lengths). Fig. 1 illustrates the main ideas of our paper.

kSeg-SVM can be efficiently trained on all available video using the SO-SVM framework [28]. Recent research [18] in the related area of sequence-labeling has shown that SO-SVMs can out-perform other algorithms including Hidden Markov Model (HMM), Conditional Random Field [11] and Max-Margin Markov Networks [24]. SO-SVMs have several benefits in the context of AU detection: (1) they model the dependencies between visual features and the duration of AUs; (2) they can be trained effectively on all possible segments of the video (rather than on independent sequences); (3) they explicitly select negative examples that are most similar to the AU to be detected; and (4) they make no assumptions about the underlying structure of the AU events (e.g., i.i.d.). Finally, we propose a novel parameterization of the output space to handle multiple AU event occurrences such that occur in long time series and search simultaneously for the k-or-fewer best matching segments in the time-series.

2. Previous work

There is a large body of research on automatic analysis of facial expressions from video. This section reviews some related work; for comprehensive surveys see [9, 19, 25, 30].

In the case of static models, different feature representations and classifiers for frame-by-frame facial expression detection have been extensively studied. Barlett et al. [3] and Littlewort et al. [12] used Gabor filters in conjunction with AdaBoost feature selection followed by an SVM classifier. Tian et al. [25] incorporated geometric features by tracking facial components. Lucey et al. [15] evaluated different shape and appearance representations derived from an AAM facial feature tracker. Zhu et al. [31] realized the importance of automatically selecting the positive and negative samples in training classifiers for AU detection. [31] showed how the AU detection can be greatly improved by automatically selecting the most discriminative training samples of those that were manually coded.

More recent work has focused on incorporating the dynamics of facial expressions to improve performance. A popular strategy is to use hidden state models to temporally segment the expression by establishing a correspondence between the action's onset, peak, and offset and an underlying latent state. Valstar and Pantic [29] used a combination of SVM and a HMM to temporally segment and recognize AUs. Koelstra and Pantic [10] used GentleBoost classifiers on motion from a non-rigid registration combined with an HMM. Similar approaches include a nonparametric discriminant HMM from Shang and Chan [21], and partially-observed Hidden Conditional Random Fields by Chang et al. [6]. Tong et al. [27] used Dynamic Bayesian Networks that make use of co-occurrence relations between AUs.

Figure 3. AAM tracking across several frames

3. Face tracking and feature extraction

This section describes the system for facial feature tracking using Active Appearance Models (AAMs) [7, 16], and the feature extraction at a frame-level. The feature representation at the segment-level is described in Sections 4.3.1 and 4.3.2.

3.1. Facial feature tracking

The facial features were tracked using a person-specific AAM model [16]. In our case, the AAM model used is composed of 66 landmarks distributed along the top of the eyebrows, the inner and outer lip outlines, the outline of the eyes, the jaw, and along the nose. Fig. 3 shows an example of AAM tracking of facial features in several frames from the RU-FACS [4] video dataset.

3.2. Feature extraction

Previous work [3, 14] has shown that appearance-based features yield good performance on many AU. In this work we follow recent work of Zhu et al. [31] and use fixed-scale-and-orientation SIFT descriptors [13] anchored at several points of interest at the tracked landmarks. Intuitively, the histogram of gradient orientations calculated in SIFT has the potential to capture much of the information that is described in FACS (e.g., the markedness of the naso-labial furrows, the direction and distribution of wrinkles, the slope of the eyebrows). At the same time, the SIFT descriptor has been shown to be robust to illumination changes and small errors in localization.

After the facial components have been tracked in each frame, a normalization step registers each image with respect to an average face [31]. An affine texture transformation is applied to each image so as to warp the texture into this canonical reference frame. This normalization provides further robustness to the effects of head motion. Once the texture is warped into this fixed reference, SIFT descriptors are computed around the outer outline of the mouth (11 points for lower face AU) and on the eyebrows (5 for upper face AU). Due to the large number of resulting features (128 by number of points), the dimensionality of the resulting feature vector was reduced using PCA to keep 95% of the energy, obtaining 261 and 126 features for lower face and upper face AU respectively.

4. Segment-based SVMs for AU localization

This section frames the AU event detection problem as a structured output learning problem.

4.1. Structured output learning

Given the frame-level features computed in the previous section, we will denote each processed video sequence i as $\mathbf{x}_i \in \mathbb{R}^{d \times m_i}$, where d is the number of features and m_i is the number of frames in the sequence. To simplify, we will assume that each sequence contains at most one occurrence of the AU event to be detected. This will be extended to k-or-fewer occurrences in Sec. 5.1. The AU event will be described by its corresponding onset to offset frame range and will be denoted by $\mathbf{y}_i \in \mathbb{Z}^2$.

Let the full training set of video sequences be $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathcal{X}$, and their associated ground truth annotations for the occurrence of AUs $\mathbf{y}_1, \cdots, \mathbf{y}_n \in \mathcal{Y}$. We wish to learn a mapping $f : \mathcal{X} \rightarrow \mathcal{Y}$ for automatically detecting the AU events in unseen signals. This complex output space contains all contiguous time intervals; each label $\mathbf{y}_i$ consists of two numbers indicating the onset and the offset of an AU:

$$\mathcal{Y} = \{\mathbf{y} \mid \mathbf{y} = \emptyset \text{ or } \mathbf{y} = [s, e] \in \mathbb{Z}^2, 1 \leq s \leq e\}. \quad (1)$$

The empty label $\mathbf{y} = \emptyset$ indicates no occurrence of the AU. We will learn the mapping f as in the structured learning framework [2, 5, 28] as:

$$f(\mathbf{x}) = \underset{\mathbf{y} \in \mathcal{Y}}{\operatorname{argmax}}\, g(\mathbf{x}, \mathbf{y}), \quad (2)$$

where $g(\mathbf{x}, \mathbf{y})$ assigns a score to any particular labeling $\mathbf{y}$; the higher this value is, the closer $\mathbf{y}$ is to the ground truth annotation. For structured output learning, the choice of $g(\mathbf{x}, \mathbf{y})$ is often taken to be a weighted sum of features in the feature space:

$$g(\mathbf{x}, \mathbf{y}) = \mathbf{w}^T \varphi(\mathbf{x}, \mathbf{y}). \quad (3)$$

where $\varphi(\mathbf{x}, \mathbf{y})$ is a joint feature mapping for temporal signal $\mathbf{x}$ and candidate label $\mathbf{y}$, and $\mathbf{w}$ is the weight vector. Learning f can therefore be posed as an optimization problem:

$$\begin{aligned} &\min_{\mathbf{w}, \boldsymbol{\xi}} \frac{1}{2}||\mathbf{w}||^2 + C \sum_{i=1}^{n} \xi_i, \quad (4) \\ &\text{s.t. } \mathbf{w}^T \varphi(\mathbf{x}_i, \mathbf{y}_i) \geq \mathbf{w}^T \varphi(\mathbf{x}_i, \mathbf{y}) + \Delta(\mathbf{y}_i, \mathbf{y}) - \xi_i \; \forall \mathbf{y}, \\ &\quad \xi_i \geq 0 \; \forall i. \end{aligned}$$

Here, $\Delta(\mathbf{y}_i, \mathbf{y})$ is a loss function that decreases as a label $\mathbf{y}$ approaches the ground truth label $\mathbf{y}_i$. Intuitively, the constraints in Eq. 4 force the score of $g(\mathbf{x}, \mathbf{y})$ to be higher for the ground truth label $\mathbf{y}_i$ than for any other value of $\mathbf{y}$, and moreover, to exceed this value by a margin equal to the loss associated with labeling $\mathbf{y}$.

4.2. Optimization and inference

The learning formulation (Eq. 4) is equivalent to:

$$\min_{\mathbf{w},\boldsymbol{\xi}} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i, \tag{5}$$
$$\text{s.t. } \max_{\mathbf{y}}\{\Delta(\mathbf{y}_i,\mathbf{y}) + \mathbf{w}^T\varphi(\mathbf{x}_i,\mathbf{y})\} \leq \mathbf{w}^T\varphi(\mathbf{x}_i,\mathbf{y}_i) + \xi_i,$$
$$\xi_i \geq 0\ \forall i.$$

This optimization problem is convex, but it has an exponentially large number of constraints. A typical optimization strategy is *constraint generation* [28] that is theoretically guaranteed to produce a global optimal solution. Constraint generation is an iterative procedure that optimizes the objective w.r.t. a smaller set of constraints. The constraint set is expanded at every iteration by adding the most violated constraint. Thus at each iteration of constraint generation, given the current value of $\mathbf{w}$, we need to solve:

$$\hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y}\in\mathcal{Y}}\{\Delta(\mathbf{y}_i,\mathbf{y}) + \mathbf{w}^T\varphi(\mathbf{x}_i,\mathbf{y})\}. \tag{6}$$

Thus for the feasibility of the training phase, it is necessary that (6) can be solved effectively and efficiently at every iteration. It is worth to note that this inference problem is different from the one for localizing AUs in a signal:

$$\hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y}\in\mathcal{Y}} \mathbf{w}^T\varphi(\mathbf{x},\mathbf{y}). \tag{7}$$

The optimization of (6) & (7) depends on the feature representation $\varphi(\mathbf{x},\mathbf{y})$. In the next section we describe two types of signal representation that render fast optimization.

4.3. Signal representation

4.3.1 Histogram of temporal words

Suppose each frame of the temporal signal is associated with a feature vector. Following [17], we consider the feature mapping $\varphi(\mathbf{x},\mathbf{y})$ as the histogram of temporal words. For quantization, we use a temporal dictionary built by applying a clustering algorithm to a set of feature vectors sampled from the training data [22]. Subsequently, each feature vector is represented by the ID of the corresponding vocabulary entry. Finally, the feature mapping $\varphi(\mathbf{x},\mathbf{y})$ is taken as the histogram of IDs associated with the frames inside the interval $\mathbf{y}$. Let $\mathbf{x}^i$ be the feature vector associated with the i^{th} frame of signal $\mathbf{x}$, and let $\mathcal{C}_j$ denote the cluster j of the temporal dictionary. The feature mapping is defined as:

$$\varphi(\mathbf{x},\mathbf{y}) = [\varphi_1,\cdots,\varphi_d,\text{len}(\mathbf{y})]^T, \tag{8}$$
$$\text{where } \varphi_j = \sum_{i\in\mathbf{y}}\varphi_j^i;\ \varphi_j^i = \delta(\mathbf{x}^i \in \mathcal{C}_j). \tag{9}$$

Here $[\varphi_1,\cdots,\varphi_d]^T$ is the histogram of temporal words located within segment $[s,e]$ of signal $\mathbf{x}$. The feature vector is the histogram concatenated with the length of the segment.

4.3.2 Soft clustering

Sec. 4.3.1 describes a feature mapping in which each frame is associated with only one cluster. In contrast with this harsh quantization, in this subsection we proposes a novel feature mapping which is based on *soft clustering*:

$$\varphi(\mathbf{x},\mathbf{y}) = [\varphi_1,\cdots,\varphi_d,\text{len}(\mathbf{y})]^T, \tag{10}$$
$$\text{where } \varphi_j = \sum_{i\in\mathbf{y}}\varphi_j^i;\ \varphi_j^i = k(\mathbf{x}^i,\mathbf{c}_j). \tag{11}$$

Here $\{\mathbf{c}_j\}$ are cluster centers, and $k(\cdot,\cdot)$ is the kernel function that measures the similarity between the frame $\mathbf{x}^i$ to the cluster center $\mathbf{c}_j$. φ_j measures the total similarity of the frames inside the segment $[s,e]$ to the cluster center $\mathbf{c}_j$.

Notably, the vectors $\{\mathbf{c}_j\}$ do not need to be the cluster centers. They can be chosen to be any set of representative vectors. For example, $\{\mathbf{c}_j\}$ can be taken as the support vectors of a frame-based SVM trained to distinguish between individual positive and negative frames. In this case, our method directly improves the performance of frame-based SVM by relearning the weights to incorporate temporal constraints. To see this, consider the score function of frame-based SVM. For a frame $\mathbf{x}^i$ of a given signal $\mathbf{x}$, the SVM score is of the form $\mathbf{v}^T\varphi(\mathbf{x}^i) + b$. It has been shown that $\mathbf{v}$ can be expressed as a linear combination of the support vectors:

$$\mathbf{v} = \sum_{j=1}^{d}\alpha_j\varphi(\mathbf{c}_j). \tag{12}$$

Thus the SVM score for frame $\mathbf{x}^i$ is:

$$\mathbf{v}^T\varphi(\mathbf{x}^i) + b = \sum_{j=1}^{d}\alpha_j k(\mathbf{x}^i,\mathbf{c}_j) + b. \tag{13}$$

Meanwhile, the decision function of structured learning is:

$$\mathbf{w}^T\varphi(\mathbf{x},\mathbf{y}) = \sum_{i=s}^{e}\sum_{j=1}^{d} w_j k(\mathbf{x}^i,\mathbf{c}_j) + w_{d+1}.\text{len}(\mathbf{y}). \tag{14}$$

Observe the similarity between the decision function of frame-based SVM and the decision function of segment-based SVM, Eq. 13 versus Eq. 14. In both cases, we need to learn a weight vector that is associated with the similarity measurement between a frame and the support vectors $\{\mathbf{c}_j\}$. Furthermore, ignoring the constant threshold, the decision value of segment-based SVM is the sum of the decision values of frame-based SVM at all frames inside the segment. The key differences between frame-based SVM and segment-based SVM are: (1) frame-based SVM classifies each frame independently while segment-based SVM makes a collective decision; (2) segment-based SVM incorporates spatial constraints during training and testing while frame-based SVM does not.

4.4. Decomposability and fast optimization

This section discusses the decomposability of the above feature mappings and how they enable efficient and effective optimization of (7) and (6).

For both feature mappings defined in Eq. 9 and Eq. 11, let a_i denote $\sum_{j=1}^{d} w_j \varphi_j^i + w_{d+1}$. Thus $\mathbf{w}^T \varphi(\mathbf{x}, \mathbf{y}) = \sum_{i=s}^{e} a_i$. The label $\hat{\mathbf{y}}$ that maximizes $\mathbf{w}^T \varphi(\mathbf{x}, \mathbf{y})$ is:

$$\hat{\mathbf{y}} = [\hat{s}, \hat{e}] = \underset{1 \le s \le e}{\operatorname{argmax}} \sum_{i=s}^{e} a_i. \tag{15}$$

There exists an efficient algorithm [17] for optimizing Eq. 15. The label $\hat{\mathbf{y}}$ that maximizes $\Delta(\mathbf{y}_i, \mathbf{y}) + \mathbf{w}^T \varphi(\mathbf{x}_i, \mathbf{y})$ can be found as:

$$\hat{\mathbf{y}} = [\hat{s}, \hat{e}] = \underset{1 \le s \le e}{\operatorname{argmax}} \{\Delta(\mathbf{y}_i, [s, e]) + \sum_{t=s}^{e} a_t\}. \tag{16}$$

For certain types of the loss function $\Delta(\mathbf{y}_i, \mathbf{y})$, Eq. 16 can be optimized very efficiently by means of a branch-and-bound algorithm [5].

5. Dealing with multiple AU occurrences

Sec. 4 presents a method to train a segment-based SVM for AU localization. It assumes the AU of interest does not occur more than once in each temporal signal. In practice, however, a temporal signal usually contains multiple AU occurrences. This raises several technical difficulties in directly applying the method presented in Sec. 4. First, even though training samples can be ensured to contain no more than one AU occurrence by breaking long training signals into smaller ones, it is unclear what the optimal division is. Second, unlike the case of training samples, we cannot ensure testing signals to contain no more than one AU occurrence. Thus it is unclear how to extend the above method to find multiple AU occurrences in a temporal signal. A possible solution is to localize multiple AU occurrences sequentially. However, sequential optimization is very likely to lead to a suboptimal solution. This section presents a method that addresses these difficulties by extending the above formulation to allow for multiple event occurrences in both training and testing stages.

5.1. Enriching the label set

To handle multiple occurrences of an AU, we propose to keep the same structured output learning formulation as in Sec. 4.1 but enriching the label set $\mathcal{Y}$. Suppose we know a priori that the number of occurrences of the AU of interest is bounded by k (such an upper bound always exists in practice). Consider the label space of all k-segmentations [17]. A k-segmentation of a time series is defined as a set of k disjoint time-intervals. Note that it is possible for some intervals of a k-segmentation to be empty. Formally speaking, the label space is defined as follows:

$$\mathcal{Y}^k = \{(I_1, \cdots, I_k) | I_i \in \mathcal{Y}, I_i \cap I_j = \phi \; \forall i, j\}. \tag{17}$$

Because the intervals of a k-segmentation can be empty, the above label set is rich enough to allow for different number of occurrences of AUs.

5.2. Inference and loss function

As in the previous case, for any fixed $\mathbf{w}$, we need efficient algorithms for the inference problems of Eq. 6 & 7. Eq. 7 can be solved efficiently because:

$$\mathbf{w}^T \varphi(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^{k} \sum_{t \in I_j} a_t. \tag{18}$$

To find $\hat{\mathbf{y}}$ that maximizes $\mathbf{w}^T \varphi(\mathbf{x}, \mathbf{y})$, we first use a linear time algorithm [17] to find:

$$\hat{\mathbf{y}} = (\hat{I}_1, \cdots, \hat{I}_k) = \underset{\substack{I_1, \ldots, I_k \\ I_i \cap I_l = \phi}}{\operatorname{argmax}} \sum_{j=1}^{k} \sum_{t \in I_j} a_t. \tag{19}$$

To take into account the imbalance of positive and negative frames, we propose to use the loss function $\Delta(\mathbf{y}_i, \mathbf{y})$ which is defined as follows:

$$\Delta(\mathbf{y}_i, \mathbf{y}) = \alpha . \text{len}(\mathbf{y}_i \setminus \mathbf{y}) + \beta . \text{len}(\mathbf{y} \setminus \mathbf{y}_i). \tag{20}$$

Here α and β are penalties for false negative and false positive frames respectively. This cost function decreases to zero when the label $\mathbf{y}$ approaches $\mathbf{y}_i$. Let

$$b_j = \begin{cases} a_j - \alpha & \text{if } j \in \mathbf{y}_i, \\ a_j + \beta & \text{otherwise.} \end{cases} \tag{21}$$

We have

$$\Delta(\mathbf{y}_i, \mathbf{y}) + \mathbf{w}^T \varphi(\mathbf{x}_i, \mathbf{y}) = \sum_{j=1}^{k} \sum_{t \in I_j} b_t + \alpha . \text{len}(\mathbf{y}_i). \tag{22}$$

Eq. 22 has the form of Eq. 18 and can be optimized similarly.

6. Experiments

This section describes experiments on two spontaneous datasets for AU detection. Experiment 1 (Sec. 6.3) compares the performance of our method against a state-of-the-art static method on a large dataset of FACS coded videos. In Experiment 2 (Sec. 6.4) we evaluate the generalization performance by testing on a dataset that was not used for training. Comparisons with dynamic models are not presented because code was not available for comparison.

6.1. Datasets and AU selection

Evaluations of performance for Experiment 1 were carried out on a relatively large corpus of FACS coded videos, the RU-FACS-1 [4] dataset. Recorded at Rutgers University, subjects were asked to either lie or tell the truth under a false opinion paradigm in interviews conducted by police and FBI members who posed around 13 questions. These interviews resulted in 2.5 minute long continuous 30-fps video sequences containing spontaneous AUs of people of varying ethnicity and sex. Ground truth FACS coding was provided by expert coders. Data from 28 of the subjects was available for our experiments. In particular, we divided this dataset into 17 subjects for training (97000 frames) and 11 subjects for testing (67000 frames).

Previously published results on this dataset (e.g., [3]) are not directly comparable due to differences in the evaluation method; for comparison purposes, we implemented a frame-based RBF SVM similar to [3].

The AU for which we present results were selected by requiring at least 100 event occurrences in the available RU-FACS-1 data, resulting in the following set of AU: $1, 2, 12, 14, 15, 17, 24$. Additionally, to test performance on AU combinations, AU1+2 was selected due to the large number of occurrences.

Experiment 2 tests generalization performance on a different dataset, *Sayette*[1]. This dataset records subjects participating in a 3-way conversation to study the effects of alcohol on social behavior. Video for 3 subjects was available to us (32000 frames) and included moderate to large head motion as subjects turned toward and away from each other, and frequent partial occlusion as subjects drank beverages. Only FACS codes for AU 6 and 12 were available. AU6 and 6+12 (which distinguishes the Duchenne smile) were therefore added to our target set.

6.2. Experimental setup and evaluation

We compare our method against a frame-based SVM and a hard-clustering BoW-kSeg approach. All methods use the same frame-level features described in Sec. 3.

The frame-based SVM is trained to distinguish between positive (AU) negative (non-AU) frames and uses a radial basis kernel $k(\mathbf{x}, \mathbf{z}) = \exp(-\gamma||\mathbf{x} - \mathbf{z}||^2)$.

The *BoW-kSeg* approach is based on a hard-clustering histogram of temporal words (Sec. 4.3.1). This approach is modeled directly after the approach used for 2D object-detection. For quantization, we use a temporal dictionary of 256 entries obtained by applying hierarchical K-means clustering to the training set feature vectors [22].

Our method (referred to as *kSeg-SVM*) is based on soft-clustering (Sec. 4.3.2). The cluster centers are chosen to be the support vectors (SVs) of frame-based SVMs with a radial basis kernel. Because for several AUs the number of SVs can be quite large $(2000 - 4000)$, we apply the idea proposed by Avidan [1] to reduce the number of SVs for faster training time and better generalization. However, instead of using a greedy algorithm for subset selection, we use LASSO regression [26]. In our experiments, the sizes of the reduced SV sets ranges from 100 to 500 SVs.

Following previous work [3], positive samples were taken to be frames where the AU was present, and negative samples where it was not (although better strategies are possible [31]). To evaluate performance, we report various measures: the area under the ROC, the precision-recall values, and the maximum $F1$ score. the $F1$ score is defined as: $F1 = \frac{2 \cdot Recall \cdot Precision}{Recall + Precision}$, summarizing the trade-off between high recall rates and accuracy among the predictions. All measures were computed on a frame-by-frame basis by varying the bias or threshold of the corresponding classifier. In our case, the $F1$ score is a better performance measure than the more common ROC metric because the latter is designed for balanced binary classification rather than detection tasks , and fails to reflect the effect of the proportion of positive to negative samples on classification performance.

Parameter tuning is done using 3-fold subject-wise cross-validation on the training data. For the frame-based SVM, we need to tune C and γ, the scale parameter of the radial basis kernel. For BoW-kSeg and kSeg-SVM, we need to tune C only. The kernel parameter γ of kSeg-SVM could also potentially be tuned, but for simplicity it was set to the same γ used for frame-based SVM. For all methods, we choose the parameters that maximize the average cross-validation ROC area.

6.3. Within dataset performance

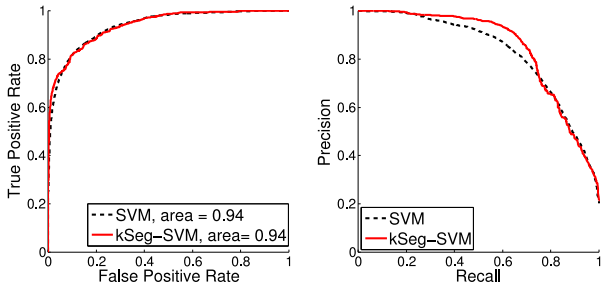
Tab. 1 shows the experimental results on the RU-FACS-1 dataset. As can be seen, kSeg-SVM consistently outperforms frame-based classification. It has the highest ROC area for 7 out of 10 AUs. Using the ROC metric, kSeg-SVM appears comparable to standard SVM. kSeg-SVM achieves highest $F1$ score on 9 out of 10 test cases.

As noted above, the $F1$ metric may be better suited for imbalanced detection tasks. Using this criterion, kSeg-SVM shows a substantial improvement. To illustrate this, consider Fig. 4 depicting ROC and precision-recall curves of AU12 and AU15. According to the ROC metric, kSeg-SVM and SVM seem comparable. However, the precision-recall curves clearly show superior performance for kSeg-SVM . For example, at 70% recall, the precision of SVM and kSeg-SVM are 0.79 and 0.87, respectively. At 50% recall for AU15, the precision of SVM is 0.48 compared to 0.67, roughly $\frac{2}{3}$ that of kSeg-SVM .

[1]This is an in-progress data-collection.

	Area under ROC			Max $F1$ score		
AU	SVM	BoW-kSeg	kSeg-SVM	SVM	BoW-kSeg	kSeg-SVM
1	**0.86**	0.52	**0.86**	0.48	0.13	**0.59**
2	0.79	0.45	**0.81**	0.42	0.14	**0.56**
6	0.89	0.69	**0.91**	0.50	0.28	**0.59**
12	**0.94**	0.77	**0.94**	0.74	0.61	**0.78**
14	**0.70**	0.56	0.68	0.20	0.17	**0.27**
15	**0.90**	0.49	**0.90**	0.50	0.04	**0.59**
17	**0.90**	0.51	0.87	0.55	0.06	**0.56**
24	**0.85**	0.52	0.73	**0.15**	0.04	0.08
1+2	0.86	0.46	**0.89**	0.36	0.12	**0.56**
6+12	0.95	0.69	**0.96**	0.55	0.28	**0.62**

Table 1. Performance on the RU-FACS-1 dataset. Higher numbers indicate better performance, and best results are printed in bold. kSeg-SVM is consistently the best according to both measures.



(a) AU 12 ROC (b) AU 12 Precision-Recall

(c) AU 15 ROC (d) AU 15 Precision-Recall

Figure 4. ROCs and precision-recall curves for AU 12 and AU 15. Although there is not a notable difference in the measured area under the ROC, precision-recall curves show a substantial improvement for our method.

As shown in Tab. 1, BoW-kSeg performs poorly. There are two possible reasons for this. First, clustering is done with K-means, an unsupervised, non-discriminative method that is not informed by the ground truth labels. Second, due to the hard dictionary assignment, each frame is forced to commit to a single cluster. While hard-clustering shows good performance in the task of object-detection, our time-series vary smoothly, resulting in large groups of consecutive frames being assigned to the same cluster.

6.4. Across dataset performance

In the second experiment we compared the generalization performance of SVM and our method across datasets. SVM and kSeg-SVM are trained on RU-FACS-1, and tested on Sayette, a separate dataset. Tab. 2 shows the ROC areas and the maximum $F1$ scores of both methods. As shown, our method kSeg-SVM consistently outperforms SVM by a large margin for all AU and their combination. Tab. 3 shows the precision values of both methods at two typical recall values of interest. The precision values of kSeg-SVM are always higher than those of SVM; in many cases the difference is as high as 50%.

	Area under ROC		Max $F1$ score	
AU	SVM	kSeg-SVM	SVM	kSeg-SVM
6	0.92	**0.94**	0.51	**0.62**
12	0.91	**0.92**	0.78	**0.79**
6+12	0.91	**0.93**	0.52	**0.61**

Table 2. Performance on the Sayette dataset. SVM and kSeg-SVM are trained on the RU-FACS-1 dataset which is a completely separated from Sayette.

	50% recall		70% recall	
AU	SVM	kSeg-SVM	SVM	kSeg-SVM
6	0.49	**0.60**	0.36	**0.54**
12	0.91	**0.95**	0.83	**0.87**
6+12	0.44	**0.56**	0.30	**0.53**

Table 3. Sayette dataset – precision values at recall values of interest.

7. Conclusions

We have extended the BoW model, which was originally proposed for spatial object detection, to the complex task of action unit detection in non-posed behavior. The kSeg-SVM demonstrated competitive performance across various AU detection tasks on a large video corpus. On measures of area under the ROC curve, our approach and frame-based SVM perform similarly. On the more challenging $F1$ measure, our method outperformed SVM for 9 out of 10 AUs.

Until now, a common measure of classifier performance for AU detection has been area under the curve. In object detection, the common measure represents the relation between recall and precision. The two approaches give very different views of classifier performance. This difference is not unanticipated in the object detection literature, but little attention has been paid to this issue in facial expression literature. Our findings underscore the importance of considering both types of measures.

In this paper we have illustrated the benefits of kSeg-SVM in the context of action unit detection, however,

the method can be applied to other domains such as action recognition and event detection in video.

Acknowledgements

Portions of this work were supported by NIMH grant MH51435. Thanks to Iain Matthews for providing the AAM tracker.

References

[1] S. Avidan. Subset selection for efficient SVM tracking. In *Computer Vision and Pattern Recognition*, 2003.

[2] G. Bakir, T. Hofmann, B. Schölkopf, A. Smola, B. Taskar, and S. Vishwanathan, editors. *Predicting Structured Data*. MIT Press, Cambridge, MA, 2007.

[3] M. Bartlett, G. Littlewort, M. Frank, C. Lainscsek, I. Fasel, and J. Movellan. Recognizing facial expression: Machine learning and application to spontaneous behavior. In *Computer Vision and Pattern Recognition*, 2005.

[4] M. Bartlett, G. Littlewort, M. Frank, C. Lainscsek, I. Fasel, and J. Movellan. Automatic recognition of facial actions in spontaneous expressions. *Journal of Multimedia*, 2006.

[5] M. B. Blaschko and C. H. Lampert. Learning to localize objects with structured output regression. In *European Conference on Computer Vision*, 2008.

[6] K. Chang, T. Liu, and S. Lai. Learning partially-observed hidden conditional random fields for facial expression recognition. In *Computer Vision and Pattern Recognition*, 2009.

[7] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. In *European Conference on Computer Vision*, 1998.

[8] P. Ekman and W. Friesen. Facial action coding system: A technique for the measurement of facial movement. *Consulting Psychologists Press*, 1978.

[9] B. Fasel and J. Luettin. Automatic facial expression analysis: a survey. *Pattern Recognition*, 36(1):259 – 275, 2003.

[10] S. Koelstra and M. Pantic. Non-rigid registration using free-form deformations for recognition of facial actions and their temporal dynamics. In *International Conference on Automatic Face and Gesture Recognition*, 2008.

[11] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*, 2001.

[12] G. Littlewort, M. Bartlett, I. Fasel, J. Susskind, and J. Movellan. Dynamics of facial expression extracted automatically from video. *Image and Vision Computing*, 24(6):615–625, 2006.

[13] D. Lowe. Object recognition from local scale-invariant features. In *Intl. Conference on Computer Vision*, 1999.

[14] P. Lucey, J. F. Cohn, S. Lucey, S. Sridharan, and K. M. Prkachin. Automatically detecting pain using facial actions. *Intl. Conference on Affective Computing and Intelligent Interaction*, 2009.

[15] S. Lucey, I. Matthews, C. Hu, Z. Ambadar, F. De la Torre, and J. Cohn. AAM derived face representations for robust facial action recognition. In *International Conference on Automatic Face and Gesture Recognition*, 2006.

[16] I. Matthews and S. Baker. Active appearance models revisited. *International Journal of Computer Vision*, 60(2):1573–1405, 2004.

[17] M. H. Nguyen, L. Torresani, F. De la Torre, and C. Rother. Weakly supervised discriminative localization and classification: a joint learning process. In *Proceedings of International Conference on Computer Vision*, 2009.

[18] N. Nguyen and Y. Guo. Comparisons of sequence labeling algorithms and extensions. In *International Conference on Machine Learning*, 2007.

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

[20] M. Pantic and L. Rothkrantz. Facial action recognition for facial expression analysis from static face images. *IEEE Transactions on Systems, Man, and Cybernetics*, pages 1449–1461, 2004.

[21] L. Shang and K. Chan. Nonparametric discriminant HMM and application to facial expression recognition. In *Conference on Computer Vision and Pattern Recognition*, 2009.

[22] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *International Conference on Computer Vision*, 2003.

[23] Y. Sun and L. Yin. Facial expression recognition based on 3D dynamic range model sequences. In *European Conference on Computer Vision*, 2008.

[24] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *Advances in Neural Information Processing Systems*. 2003.

[25] Y. Tian, J. F. Cohn, and T. Kanade. Facial expression analysis. In S. Z. Li and A. K. Jain, editors, *Handbook of face recognition*. New York, New York: Springer, 2005.

[26] R. Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society, Series B*, 58(267–288), 1996.

[27] Y. Tong, W. Liao, and Q. Ji. Facial action unit recognition by exploiting their dynamic and semantic relationships. *Transactions on Pattern Analysis and Machine Intelligence*, pages 1683–1699, 2007.

[28] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

[29] M. Valstar and M. Pantic. Combined support vector machines and hidden markov models for modeling facial action temporal dynamics. In *ICCV Workshop on Human Computer Interaction*, 2007.

[30] Z. Zeng, M. Pantic, G. I. Roisman, and T. S. Huang. A survey of affect recognition methods: Audio, visual, and spontaneous expressions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(1):39–58, 2009.

[31] Y. Zhu, F. De la Torre, and J. Cohn. Dynamic cascades with bidirectional bootstrapping for spontaneous facial action unit detection. In *Affective Computing and Intelligent Interaction ACII*, September 2009.