# DETECTOR OF FACIAL LANDMARKS LEARNED BY THE STRUCTURED OUTPUT SVM

Michal Uřičář, Vojtěch Franc and Václav Hlaváč

*Center for Machine Perception, Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague, Technická 2, 166 27 Prague 6, Czech Republic*

Keywords: Facial Landmark Detection, Structured Output Classification, Support Vector Machines, Deformable Part Models.

Abstract: In this paper we describe a detector of facial landmarks based on the Deformable Part Models. We treat the task of landmark detection as an instance of the structured output classification problem. We propose to learn the parameters of the detector from data by the Structured Output Support Vector Machines algorithm. In contrast to the previous works, the objective function of the learning algorithm is directly related to the performance of the resulting detector which is controlled by a user-defined loss function. The resulting detector is real-time on a standard PC, simple to implement and it can be easily modified for detection of a different set of landmarks. We evaluate performance of the proposed landmark detector on a challenging "Labeled Faces in the Wild" (LFW) database. The empirical results demonstrate that the proposed detector is consistently more accurate than two public domain implementations based on the Active Appearance Models and the Deformable Part Models. We provide an open-source implementation of the proposed detector and the manual annotation of the facial landmarks for all images in the LFW database.

## 1 INTRODUCTION

The detection of facial landmarks like canthi, nose and mouth corners (see Figure 1) is an essential part of face recognition systems. The accuracy of the detection significantly influences its final performance (Beumer and Veldhuis, 2005; Cristinacce et al., 2004; Riopka and Boult, 2003). The problem of the precise and robust detection of facial landmarks has received a lot of attention in the past decade. We briefly review only the approaches relevant to the method proposed in this paper.

Among the most popular are detectors based on the Active Appearance Models (AAM) (Cootes et al., 2001) which use a joint statistical model of appearance and shape. Detectors build on AAM provide a dense set of facial features, allowing to extract whole contours of facial parts like eyes, etc. However high resolution images are required for both training and testing stage and the detection leads to solving a non-convex optimization problem susceptible to local optima unless a good initial guess of the landmark positions is available.

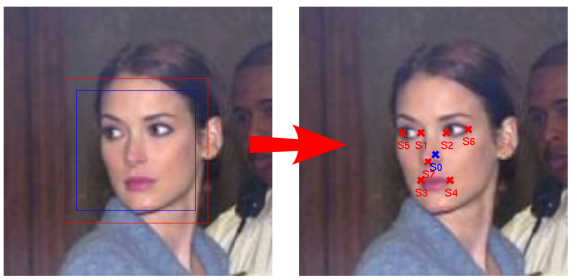



Figure 1: Functionality of the facial landmark detector.

A straightforward approach to landmark detection is based on using independently trained detectors for each facial landmark. For instance the AdaBoost based detectors and its modifications have been frequently used (Viola and Jones, 2004). If applied independently, the individual detectors often fail to provide a robust estimate of the landmark positions. The weakness of the local evidence can be compensated by using a prior on the geometrical configuration of landmarks. The detection is typically carried out in two consecutive steps. In the first step, the individual detectors are used to find a set of candidate positions for each landmark separately. In the second step, the best landmark configuration with the highest support from the geometrical prior is selected. The landmark detectors based on this approach were pro-

Uřičař M., Franc V. and Hlaváč V..
DETECTOR OF FACIAL LANDMARKS LEARNED BY THE STRUCTURED OUTPUT SVM.
DOI: 10.5220/0003863705470556
In *Proceedings of the International Conference on Computer Vision Theory and Applications* (VISAPP-2012), pages 547-556
ISBN: 978-989-8565-03-7

posed for example in (Beumer et al., 2006; Cristinacce and Cootes, 2003; Erukhimov and Lee, 2008; Wu and Trivedi, 2005).

The Deformable Part Models (DPM) (Crandall et al., 2005; Felzenszwalb and Huttenlocher, 2005; Felzenszwalb et al., 2009; Fischler and Elschlager, 1973) go one step further by fusing the local appearance model and the geometrical constraints into a single model. The DPM is given by a set of parts along with a set of connections between certain pairs of parts arranged in a deformable configuration. A natural way how to describe the DPM is an undirected graph with vertices corresponding to the parts and edges representing the connections between the pairs of connected parts. The DPM detector estimates all landmark positions simultaneously by optimizing a single scoring function composed of a local appearance model and a deformation cost. The complexity of finding the best landmark configuration depends on the structure of underlying graph. Acyclic graph allows efficient estimation by a variant of the Dynamic Programming (DP).

An instance of finely tuned facial landmark detector based on the DPM has been proposed in (Everingham et al., 2006). The very same detector was also used in several successful face recognition systems described in (Everingham et al., 2009) and (Sivic et al., 2009). In this case, the local appearance model is learned by a multiple-instance variant of the AdaBoost algorithm with Haar-like features used as the weak classifiers. The deformation cost is expressed as a mixture of Gaussian trees whose parameters are learned from examples. This landmark detector is publicly available and we use it for comparison with our detector [1]. Importantly, learning of the local appearance model and the deformation cost is done in two independent steps which simplifies learning, but may not be optimal in terms of detectors accuracy.

We propose to learn the parameters of the DPM discriminatively in one step by directly optimizing accuracy of the resulting detector. The main contributions of this paper are as follows:

1. We treat the landmark detection with the DPM as an instance of the structured output classification problem whose detection accuracy is measured by a loss function natural for this application. We propose to use the Structured Output SVM (SO-SVM) (Tsochantaridis et al., 2005) for supervised learning of the parameters of the landmark detector from examples. The learning objective of the SO-SVMs is directly related to the accuracy of the detector. In contrast, all existing approaches we are aware of optimize surrogate objective functions whose relation to the detector accuracy is not always clear.
2. We empirically evaluate accuracy of the proposed landmark detector learned by the SO-SVMs on a challenging "Labeled Faces in the Wild" database (Huang et al., 2007).
3. We provide an empirical comparison of two popular optimization algorithms — the Bundle Method for Regularized Risk Minimization (BMRM) (Teo et al., 2010) and the Stochastic Gradient Descend (SGD) (Bordes et al., 2009) — which are suitable for solving the convex optimization problem emerging in the SO-SVM learning.
4. We provide an open source library which implements the proposed detector and the algorithm for supervised learning of its parameters. In adidtion we provide a manual annotation of the facial landmarks for all images from the LFW database.

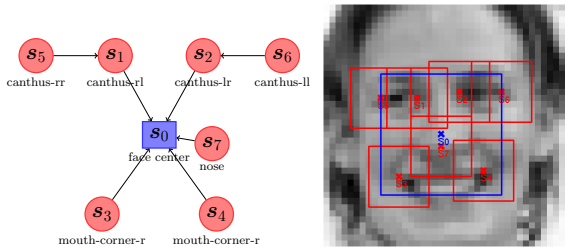
The paper is organized as follows. Section 2 defines the structured output classifier for facial landmark detection based on the DPM. Section 3 describes the SO-SVM algorithm for learning the parameters of the classifier from examples. Experimental results are presented in Section 4. Section 5 shortly describes the open source implementation of our detector and the provided manual annotation of the LFW database. Section 6 concludes the paper.

## 2 THE STRUCTURED OUTPUT CLASSIFIER FOR FACIAL LANDMARK DETECTION

We treat the landmark detection as an instance of the structured output classification problem. We assume that the input of our classifier is a still image of fixed size containing a single face. In our experiments we construct the input image by cropping a window around a bounding box found by a face detector (enlarged by a fixed margin ensuring the whole face is contained) and normalizing its size. The classifier output are estimated locations of a set of facial landmarks. A formal definition is given next.

Let $\mathcal{I} = \mathcal{X}^{H \times W}$ be a set of input images with $H \times W$ pixels where $\mathcal{X}$ denotes a set of pixel values which in our experiments, dealing with 8bit grayscale images, is $\mathcal{X} = \{0, \ldots, 255\}$. We describe the configuration of $M$ landmarks by a graph $G = (V, E)$, where $V = \{0, \ldots, M-1\}$ is a set of landmarks and

[1] There also exists a successful commercial solution OKAO Vision Facial Feature Extraction API (http://www.omron.com) which is used for example in Picasa™ or Apple iPhoto™ software.




(a) (b)

Figure 2: Definition of (a) the underlying graph $G = (V,E)$ for the landmark configuration and (b) the components of the proposed detector.

$E \subset V^2$ is a set of edges defining the neighbouring landmarks. Each landmark is assigned a position $s_i \in \mathcal{S}_i \subset \{1,\dots,H\} \times \{1,\dots,W\}$ where $\mathcal{S}_i$ denotes a set of all admissible positions of the $i$-th landmark within the image $I \in \mathcal{J}$. The quality of a landmark configuration $\mathbf{s} = (\mathbf{s}_0,\dots,\mathbf{s}_{M-1}) \in \mathcal{S} = \mathcal{S}_0 \times \cdots \times \mathcal{S}_{M-1}$ given an input image $I \in \mathcal{J}$ is measured by a scoring function $f\colon \mathcal{J} \times \mathcal{S} \to \mathbb{R}$ defined as

$$f(I,\mathbf{s}) = \sum_{i \in V} q_i(I,\mathbf{s}_i) + \sum_{(i,j)\in E} g_{ij}(\mathbf{s}_i,\mathbf{s}_j)\,. \qquad (1)$$

The first term in (1) corresponds to a local appearance model evaluating the match between landmarks on positions $\mathbf{s}$ and the input image $I$. The second term in (1) is the deformation cost evaluating the relative positions of the neighboring landmarks $i$ and $j$.

We assume that the costs $q_i\colon \mathcal{J} \times \mathcal{S}_i \to \mathbb{R}, i = 0,\dots,M-1$ and $g_{ij}\colon \mathcal{S}_i \times \mathcal{S}_j \to \mathbb{R}, (i,j) \in E$ are linearly parametrized functions

$$q_i(I,\mathbf{s}_i) = \langle \mathbf{w}_i^q, \Psi_i^q(I,\mathbf{s}_i)\rangle \qquad (2)$$
$$g_{ij}(\mathbf{s}_i,\mathbf{s}_j) = \langle \mathbf{w}_{ij}^g, \Psi_{ij}^g(\mathbf{s}_i,\mathbf{s}_j)\rangle, \qquad (3)$$

where $\Psi_i^q\colon \mathcal{J} \times \mathcal{S}_i \to \mathbb{R}^{n_{iq}}$, $\Psi_{ij}^g\colon \mathcal{S}_i \times \mathcal{S}_j \to \mathbb{R}^{n_{ig}}, i = 0,\dots,M-1$ are predefined maps and $\mathbf{w}_i^q \in \mathbb{R}^{n_{iq}}, \mathbf{w}_{ij}^g \in \mathbb{R}^{n_{ig}}, i = 0,\dots,M-1$ are parameter vectors which will be learned from examples. Let us introduce a joint map $\Psi\colon \mathcal{J} \times \mathcal{S} \to \mathbb{R}^n$ and a joint parameter vector $\mathbf{w} \in \mathbb{R}^n$ defined as a column-wise concatenation of the individual maps $\Psi_i^q, \Psi_{ij}^g$ and the individual parameter vectors $\mathbf{w}_i^q, \mathbf{w}_{ij}^g$ respectively. With these definitions we see that the scoring function (1) simplifies to $f(I,\mathbf{s}) = \langle \mathbf{w}, \Psi(I,\mathbf{s})\rangle$.

Given an input image $I$, the structured output classifier returns the configurations $\hat{\mathbf{s}}$ computed by maximizing the scoring function $f(I,\mathbf{s})$, i.e.

$$\hat{\mathbf{s}} \in \operatorname*{argmax}_{\mathbf{s}\in\mathcal{S}} f(I,\mathbf{s})\,. \qquad (4)$$

We assume that the graph $G = (V,E)$ is acyclic (see Figure 2(a)), which allows efficient solving of the maximization problem (4) by dynamic programming.

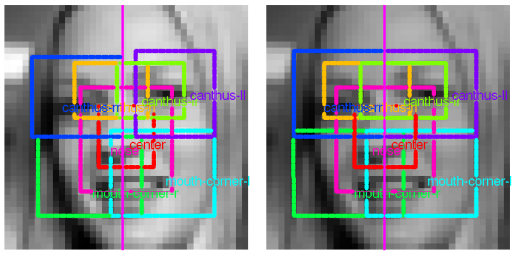


Figure 3: Left: optimal search spaces for each component. Right: the same search spaces made symmetrical along the vertical magenta line.

A complete specification of the structured classifier (4) requires to define:

- The maps $\Psi_i^q(I,\mathbf{s}_i), i = 0,\dots,M-1$ where $\Psi_i^q(I,\mathbf{s}_i)$ defines a local feature descriptor of $i$-th landmark computed on a rectangular window centered at $\mathbf{s}_i$. We call the rectangular window a component (see Figure 2(b)). The size of the component and its feature descriptor are crucial design options which have to be made carefully. In Section 2.1 we describe a list of feature descriptors we have considered.
- The fixed maps $\Psi_{ij}^g(\mathbf{s}_i,\mathbf{s}_j), (i,j) \in E$ defining the parametrization of the deformation cost. Section 2.2 describes the parametrization which we have considered.
- The set $\mathcal{S} = (\mathcal{S}_0 \times \cdots \times \mathcal{S}_{M-1})$ defining the search space of the landmark positions. These sets can be interpreted as hard constraints on the admissible configurations of the landmarks, i.e. the landmark positions outside these sets corresponds to $-\infty$ value of the deformation cost $g_{ij}(\mathbf{s}_i,\mathbf{s}_j)$.

  We tune the size of these search spaces experimentally — we keep track of the axis aligned bounding box (AABB) for each component trough the whole database excluding the images whose components does not fit in the image. We set the size of components in order to keep at least 95% images of the original database. Consequently, the AABB of each component is made vertically symmetric along the center $y$-axis in order to remove bias to certain positions. Figure 3 visualizes the found search spaces.
- The joint parameter vector $\mathbf{w} \in \mathbb{R}^n$ learned from the training examples by the SO-SVM algorithm described in Section 3.

Finally we would like to stress that the particular number of landmarks and their neighborhood structure can be arbitrary as long as the inference problem (1) can be solved efficiently. In this paper we experiment with the 8-landmarks variant of the graph

$G = (V, E)$ shown in Figure 2(a).

### 2.1 Appearance Model

We have experimented with several feature descriptors $\Psi_i^q$ for the local appearance model $q_i(I, \mathbf{s}_i)$. In particular, we considered i) normalized intensity values, ii) derivatives of image intensity values, iii) histograms of Local Binary Patterns (LBP) (Heikkilä et al., 2009) and iv) the LBP pyramid feature descriptor (Franc and Sonnenburg, 2010). We obtained the best results with the LBP pyramid feature descriptor which is used in the experiments. The LBP pyramid descriptor is constructed by concatenating binary encoded LBP features computed in each pixel (up to boundary pixels) and in several scales. In particular, we use the LBP pyramid computed in 4 scales starting from the original image and consequently downscaling the image 3 times by $1/2$. The resulting feature vector is high dimensional but very sparse.

### 2.2 Deformation Cost

We have experimented with two parametrizations of the deformation cost $g_{ij}(\mathbf{s}_i, \mathbf{s}_j)$: i) a table representation and ii) a quadratic function of a displacement vector between landmark positions.

The table representation is the most generic form of the deformation cost useful when no prior knowledge is available. Table elements specify cost for each combination of $\mathbf{s}_i$ and $\mathbf{s}_j$ separately. $\Psi_{ij}^g(\mathbf{s}_i, \mathbf{s}_j)$ is a sparse vector with all elements zero but the element corresponding to the combinations $(\mathbf{s}_i, \mathbf{s}_j)$ which is one. Though the table representation is very flexible its main disadvantage is a very large number of parameters to be learned. In turn, a large number of training examples is required to avoid over-fitting.

As the second option, we considered the deformation cost $g_{ij}(\mathbf{s}_i, \mathbf{s}_j)$ to be a quadratic function of a displacement vector $\mathbf{s}_j - \mathbf{s}_i$. Following (Felzenszwalb et al., 2009), we define the deformation cost as

$$\left.\begin{array}{rcl} \Psi_{ij}^g(\mathbf{s}_i, \mathbf{s}_j) & = & (dx, dy, dx^2, dy^2) \\ (dx, dy) & = & (x_j, y_j) - (x_i, y_i) \end{array}\right\} \quad (5)$$

This representation accounts for the distance and the direction of the $j$-th landmark with respect to $i$-th landmark. This representation is determined only by four parameters which substantially reduces the risk of over-fitting.

We found experimentally the quadratic deformation cost to give slightly better results compared to the table representation.

## 3 LEARNING THE PARAMETERS OF THE STRUCTURED OUTPUT CLASSIFIER

We learn the joint parameter vector $\mathbf{w}$ by the SO-SVM algorithm (Tsochantaridis et al., 2005). The requirements on the classifier are specified by a user defined loss-function $L\colon \mathcal{S} \times \mathcal{S} \to \mathbb{R}$. The value $L(\mathbf{s}, \mathbf{s}^*)$ penalizes the classifier estimate $\mathbf{s}$ provided the actual configuration of the landmarks is $\mathbf{s}^*$. The SO-SVM requires loss function to be non-negative and zero iff the estimate is absolutely correct, i.e. $L(\mathbf{s}, \mathbf{s}^*) \geq 0$, $\forall \mathbf{s}, \mathbf{s}^* \in \mathcal{S}$, and $L(\mathbf{s}, \mathbf{s}^*) = 0$ iff $\mathbf{s} = \mathbf{s}^*$. In particular, we use the mean normalized deviation between the estimated and the ground truth positions as the loss function, i.e.,

$$L(\mathbf{s}, \mathbf{s}^*) = \kappa(s^*) \frac{1}{M} \sum_{j=0}^{M-1} \|\mathbf{s}_j - \mathbf{s}_j^*\|. \quad (6)$$

The normalization factor $\kappa(s^*) = \| \frac{1}{2}(\mathbf{s}_{\text{eyeR}} + \mathbf{s}_{\text{eyeL}}) - \mathbf{s}_{\text{mouth}} \|^{-1}$ is reciprocal to the face size which we define as the length of the line connecting the midpoint between the eye centers $\mathbf{s}_{\text{eyeR}}$ and $\mathbf{s}_{\text{eyeL}}$ with the mouth center $\mathbf{s}_{\text{mouth}}$. The normalization factor is introduced in order to make the loss function scale invariant which is necessary because responses of the face detector used to construct the input images do not allow accurate estimation of the scale. Figure 4 illustrates the meaning of the loss function (6). Finally, we point out that any other loss function meeting the constraints defined above can be readily used, e.g. one can use maximal normalized deviation.

Given a set of training examples $\{(I^1, \mathbf{s}^1), \ldots, (I^m, \mathbf{s}^m)\} \in (\mathcal{I} \times \mathcal{S})^m$ composed of pairs of the images and their manual annotations, the parameter $\mathbf{w}$ of the classifier (4) is obtained by solving the following convex minimization problem

$$\mathbf{w}^* = \operatorname*{argmin}_{\mathbf{w} \in \mathbb{R}^n} \left[ \frac{\lambda}{2} \|\mathbf{w}\|^2 + R(\mathbf{w}) \right], \text{ where} \quad (7)$$

$$R(\mathbf{w}) = \frac{1}{m} \sum_{i=1}^{m} \max_{\mathbf{s} \in \mathcal{S}} \left( L(\mathbf{s}^i, \mathbf{s}) + \langle \mathbf{w}, \Psi(I^i, \mathbf{s}) \rangle \right) - \frac{1}{m} \sum_{i=1}^{m} \langle \mathbf{w}, \Psi(I^i, \mathbf{s}^i) \rangle . \quad (8)$$

The number $\lambda \in \mathbb{R}^+$ is a regularization constant whose optimal value is tuned on a validation set. $R(\mathbf{w})$ is a convex piece-wise linear upper bound on the empirical risk $\frac{1}{m} \sum_{i=1}^{m} L(\mathbf{s}^i, \arg\max_{\mathbf{s} \in \mathcal{S}} f(I^i, \mathbf{s}))$. That is, the learning algorithm directly minimizes the performance of the detector assessed on the training set and at the same time it controls the risk of over-fitting via the norm of the parameter vector.

Though the problem (7) is convex its solving is hard. The hardness of the problem can be seen when it is expressed as an equivalent quadratic program with $m|\mathcal{S}|$ linear constrains (recall that $|\mathcal{S}|$ is the number of all landmark configurations). This fact rules out off-the-shelf optimization algorithms.

Thanks to its importance a considerable effort has been put to a development of efficient optimization algorithms for solving the task (7). There has been an ongoing discussion in the machine learning community trying to decide whether approximative online solvers like the SGD are better than the accurate slower solvers like the BMRM. No definitive consensus has been achieved so far. We contribute to this discussion by providing an empirical evaluation of both approaches on the practical large-scale problem required to learn the landmark detector. The empirical results are provided in section 4.5. For the sake of self-consistency, we briefly describe the considered solvers, i.e. the BMRM and the SGD algorithm, in the following two sections.

## 3.1 Bundle Methods for Regularized Risk Minimization

The BMRM is a generic method for minimization of regularized convex functions (Teo et al., 2010), i.e. BMRM solves the following convex problem

$$\mathbf{w}^* = \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^n} F(\mathbf{w}) := \frac{\lambda}{2}\|\mathbf{w}\|^2 + R(\mathbf{w}),$$

where $R\colon \mathbb{R}^n \to \mathbb{R}$ is an arbitrary convex function. The risk term $R(\mathbf{w})$ is usually the complex part of the objective function which makes the optimization task hard. The core idea is to replace the original problem by its reduced problem

$$\mathbf{w}_t = \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^n} F_t(\mathbf{w}) := \frac{\lambda}{2}\|\mathbf{w}\|^2 + R_t(\mathbf{w}). \qquad (9)$$

The objective function $F_t(\mathbf{w})$ of the reduced problem (9) is obtained after replacing the risk $R(\mathbf{w})$ in the original objective $F(\mathbf{w})$ by its cutting plane model

$$R_t(\mathbf{w}) = \max_{i=0,1,\ldots,t-1} \left[R(\mathbf{w}_i + \langle R'(\mathbf{w}_i), \mathbf{w}-\mathbf{w}_i\rangle\right], \qquad (10)$$

where $R'(\mathbf{w}_i) \in \mathbb{R}^n$ denotes a subgradient of $R(\mathbf{w})$ evaluated at the point $\mathbf{w}_i \in \mathbb{R}^n$.

Starting from an initial guess $\mathbf{w}_0 = \mathbf{0}$, the BMRM algorithm computes a new iterate $\mathbf{w}_t$ by solving the reduced problem (9). In each iteration $t$, the cutting plane model (10) is updated by a new cutting plane computed at the intermediate solution $\mathbf{w}_t$ leading to a progressively tighter approximation of $F(\mathbf{w})$. The BMRM algorithm halts if the gap between $F(\mathbf{w}_t)$ (an upper bound on $F(\mathbf{w}^*)$) and $F_t(\mathbf{w}_t)$ (a lower bound on $F(\mathbf{w}^*)$) falls below a desired $\varepsilon$, meaning that $F(\mathbf{w}_t) \leq F(\mathbf{w}^*) + \varepsilon$. The BMRM algorithm halts after at most $O(1/\varepsilon)$ iterations for arbitrary $\varepsilon > 0$ (Teo et al., 2010).

The reduced problem (9) can be expressed as an equivalent convex quadratic program with $t$ variables. Because $t$ is usually small (up to a few hundreds), off-the-shelf QP solvers can be used.

Before applied to a particular problem, the BMRM algorithm requires a procedure which for a given $\mathbf{w}$ returns the value of the risk $R(\mathbf{w})$ and its sub-gradient $R'(\mathbf{w})$. In our case the risk $R(\mathbf{w})$ is defined by (8) and its sub-gradient can be computed by the Danskin's theorem as

$$R'(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m}\left(\Psi(I^i,\hat{\mathbf{s}}^i) - \Psi(I^i,\mathbf{s}^i)\right), \qquad (11)$$

$$\hat{\mathbf{s}}^i = \operatorname*{arg\,max}_{\mathbf{s}\in\mathcal{S}}\left[L(\mathbf{s}^i,\mathbf{s}) + \langle \mathbf{w}, \Psi(I^i,\mathbf{s})\rangle\right] \qquad (12)$$

Note that the evaluation of $R(\mathbf{w})$ and $R'(\mathbf{w})$ is dominated by the computation of the scalar products $\langle \mathbf{w}, \Psi(I^i,\mathbf{s})\rangle$, $i = 1,\ldots,m$, $\mathbf{s} \in \mathcal{S}$, which, fortunately, can be efficiently parallelized.

## 3.2 Stochastic Gradient Descent

Another popular method solving (7) is the Stochastic Gradient Descent (SGD) algorithm. We use the modification proposed in (Bordes et al., 2009) which uses two neat tricks. Starting from an initial guess $\mathbf{w}_0$, the SGD algorithm iteratively changes $\mathbf{w}$ by applying the following rule:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \frac{\lambda^{-1}}{t_0+t} g_t, \quad g_t = \lambda\mathbf{w}_t + \mathbf{h}_t \qquad (13)$$

$t_0$ is a constant and $t$ is the number of the iteration. The SGD implementation proposed in (Bordes et al., 2009) tunes the optimal value of $t_0$ on a small portion of training examples sub-sampled from training set. The sub-gradient is computed in almost the same manner as in (11), but only for one training image at a time, i.e., $\mathbf{h}_t = \Psi(I^t,\hat{\mathbf{s}}^t) - \Psi(I^t,\mathbf{s}^t)$.

In addition, (Bordes et al., 2009) propose to exploit the sparsity of the data in the update step. The equation (13) can be expressed as

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha_t\mathbf{w}_t - \beta_t\mathbf{h}_t, \text{ where} \qquad (14)$$

$$\alpha = \frac{1}{t_0+t}, \quad \beta = \frac{\lambda^{-1}}{t_0+t} \qquad (15)$$

Note that if $\mathbf{h}_t$ is sparse then subtracting $\beta_t\mathbf{h}_t$ involves only the nonzero coefficients of $\mathbf{h}_t$, but subtracting $\alpha_t\mathbf{w}_t$ involves all coefficients of $\mathbf{w}_t$. In turn, it is beneficial to reformulate the equation (14) as

$$\mathbf{w}_{t+1} = (1-\alpha_t)\mathbf{w}_t - \beta_t\mathbf{h}_t. \qquad (16)$$

By using this trick, the complexity $O(d)$ corresponding to the naïve implementation of the update rule (13) reduces to the complexity $O(d_{\text{non-zero}})$ corresponding to the reformulated rule (16), where $d$ is the dimension of the parameter vector and $d_{\text{non-zero}}$ is the number of the non-zero elements in $\mathbf{h}_t$. Typically, like in our case, $d_{\text{non-zero}}$ is much smaller than $d$.

A considerable advantage of the SGD algorithm is its simplicity. A disadvantage is that the SGD algorithm does not provide any certificate of optimality and thus theoretically grounded stopping condition is not available.

# 4 EXPERIMENTS

In this section, we present experimental evaluation of the proposed facial landmark detector and its comparison against three different approaches. We considered the detector estimating positions of the eight landmarks: the canthi of the left and the right eye, the corners of the mouth, the tip of the nose and the center of the face. The corresponding graph $(V,E)$ is shown in Figure2(a).

In section 4.1, we describe the face database and the testing protocol used in the experiments. The competing methods are summarized in Section 4.2. The results of the comparison in terms of detection accuracy and basic timing statistics are presented in Section 4.4. Finally, in Section 4.5 we compare two algorithms for solving the large-scale optimization problems emerging in the SO-SVM learning, namely, the BMRM and the SGD algorithm.

## 4.1 Database and Testing Protocol

We use the Labeled Faces in the Wild (LFW) database (Huang et al., 2007) for evaluation as well as for training of our detector. This database consists of $13,233$ images each of size $250 \times 250$ pixels. The LFW database contains a great ethnicity variance and the images have challenging background clutter. We augmented the original LFW database by adding manual annotation of the eight considered landmarks.

We randomly split the LFW database into training, testing and validation sets. Table 1 describes this partitioning. The experimental evaluation of all competing detectors was made on the same testing set. The training and the validation parts were used for learning of the proposed detector and the base line SVM detector. The other competing detectors had their own training databases.

In order to evaluate the detectors, we use two accuracy measures: i) the mean normalized deviation

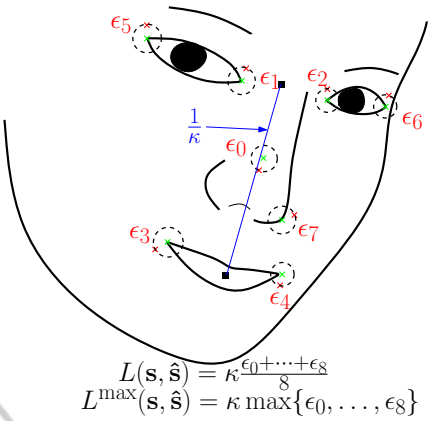



Figure 4: The illustration of two accuracy statistics used to benchmark the detectors. The green and the red crosses denote the manually annotated landmarks and the detected landmarks, respectively. The deviations $\varepsilon_0,\ldots,\varepsilon_7$ correspond to radii of the dashed circles.

Table 1: The partitioning of the LFW database into training, validation and testing set.

| Data set | Training | Validation | Testing |
|---|---|---|---|
| Percentage | 60% | 20% | 20% |
| # of examples | 6,919 | 2,307 | 2,316 |

$L(\mathbf{s},\mathbf{s}')$ defined by equation (6) and ii) the maximal normalized deviation

$$L^{\max}(\mathbf{s},\hat{\mathbf{s}}) = \kappa(\mathbf{s}) \max_{j=0,\ldots,M-1} \|\mathbf{s}_j - \hat{\mathbf{s}}_j\|, \qquad (17)$$

where $\mathbf{s} = (\mathbf{s}_0,\ldots,\mathbf{s}_{M-1})$ are the manually annotated landmark positions and $\hat{\mathbf{s}} = (\hat{\mathbf{s}}_0,\ldots,\hat{\mathbf{s}}_{M-1})$ are the landmark positions estimated by the tested detector. Figure 4 illustrates both accuracy measures.

## 4.2 Competing Methods

In this section, we outline all detectors that were used in the experimental evaluation.

### 4.2.1 Proposed Detector

The proposed detector estimates the landmark positions according to the formula (4). As the feature descriptor $\Psi_i^q(I,\mathbf{s}_i)$ defining the local appearance model $q_i(I,\mathbf{s}_i)$, we use the LBP pyramid described in Section 2.1. As the parametrization $\Psi_{ij}^g(\mathbf{s}_i,\mathbf{s}_j)$ of the deformation cost $g_{ij}(\mathbf{s}_i,\mathbf{s}_j)$, we use the quadratic function described in Section 2.2. The parameter vector $\mathbf{w}$ of the classifier (4) is trained from the training part of the LFW database using the BMRM algorithm (c.f. Section 3.1). The regularization constant $\lambda$ appearing in the learning problem (7) was selected from

the set $\{10, 1, 0.1, 0.01, 0.001\}$ to minimize the average mean normalized deviation $R_{\text{VAL}}$ computed on the validation part of the LFW database.

#### 4.2.2 Independently Trained SVM Detector

This detector is formed by standard two-class linear SVM classifiers trained independently for each landmark. For training, we use the SVM solver implemented in LIBOCAS (Franc and Sonnenburg, 2010). For each individual landmark we created a different training set containing examples of the positive and negative class. The positive class is formed by images cropped around the ground truth positions of the respective component. The negative class contains images cropped outside the ground truth regions. Specifically, the negative class images satisfy the following condition

$$\left|P^x_- - P^x_{\text{GT}}\right| > \frac{1}{2}\text{width}_{\text{GT}},\ \left|P^y_- - P^y_{\text{GT}}\right| > \frac{1}{2}\text{height}_{\text{GT}}$$

where $P^x_-$ and $P^x_{\text{GT}}$ is the $x$-coordinate of the negative and the ground truth component respectively. $\text{height}_{\text{GT}}$ and $\text{width}_{\text{GT}}$ denote the width and the height of the component.

We use the LBP-pyramid descriptor (see Section 2.1) as the features. The parameters of the linear SVM classifier are learned from the training part of the LFW database. The SVM regularization constant $C$ was selected from the set $\{10, 1, 0.1, 0.01, 0.001\}$ to minimize the classification error computed on the validation part of the LFW database.

Having the binary SVM classifiers trained for all components, the landmark position is estimated by selecting the place with the maximal response of the classifier scoring function, evaluated in the search regions defined for each component differently. The search regions as well as the sizes of the components are exactly the same as we use for the proposed SO-SVM detector.

Note that the independently trained SVM detector is a simple instance of the DPM where the deformation cost $g_{ij}(\mathbf{s}_i, \mathbf{s}_j)$ is zero for all positions inside the search region and $-\infty$ outside. We compare this baseline detector with the proposed SO-SVM detector to show that by learning the deformation cost from data one can improve the accuracy.

#### 4.2.3 Active Appearance Models

We use a slightly modified version of a publicly available implementation of the AAM (Kroon, 2010). As the initial guess of the face position required by the AAM, we use the center of the bounding box obtained from a face detector. The initial scale is also computed from this bounding box. The AAM estimates a dense set of feature points which are distributed around important face contours like the contour of mouth, eyes, nose, chin and eyebrows. The AAM requires a different training database which contains high resolution images along with annotation of all contour points.

For training the AAM model we use a publicly available IIM Face database (Nordstrøm et al., 2004). The IIM database consists of 240 annotated images (6 images per person). Each image is $640 \times 480$ pixel in size and comes with 58 manually annotated points which are distributed along the main face contours. Note that the creation of training examples for the AAM put much higher demands on the annotator—he/she has to click a large number of uniformly distributed points. In contrast, our method requires annotation of only a small number of well defined points. Specifically, the whole IIM database requires to annotate 13920 points, carefully distributed along each contour, while the LFW database requires to annotate 48433 points, which are well defined and easy to annotate.

To compare the AAM based detector with our detector, we have to transform the output of the AAM, i.e. the points on contours around important face parts, to the landmark positions returned by our detector. We simply select the relevant points on contours.

### 4.3 Detector of Everingham et al.

The last competing detector is the DPM based detector of (Everingham et al., 2008). This detector was trained on a collection of consumer images which, however, are not available. This detector returns canthi of both eyes (4 landmarks), corners of the mouth (2 landmarks) and 3 landmarks on the nose. To compare this detector, we consider only the relevant landmarks for our detector. Note that unlike the proposed SO-SVM detector, this detector learns the local appearance model and the deformation cost of the DPM independently.

### 4.4 Results

In this section, we describe results of the experimental evaluation of the detection accuracy of all competing detectors. We have measured the mean and the maximal normalized deviation computed on the test part of the LFW database.

Table 2 shows the average mean normalized deviation $R_{\text{TST}}$ and the the average maximal normalized deviation $R^{\max}_{\text{TST}}$ for each individual detector. The results show that the proposed detector consistently outperforms all other competing methods irrespective

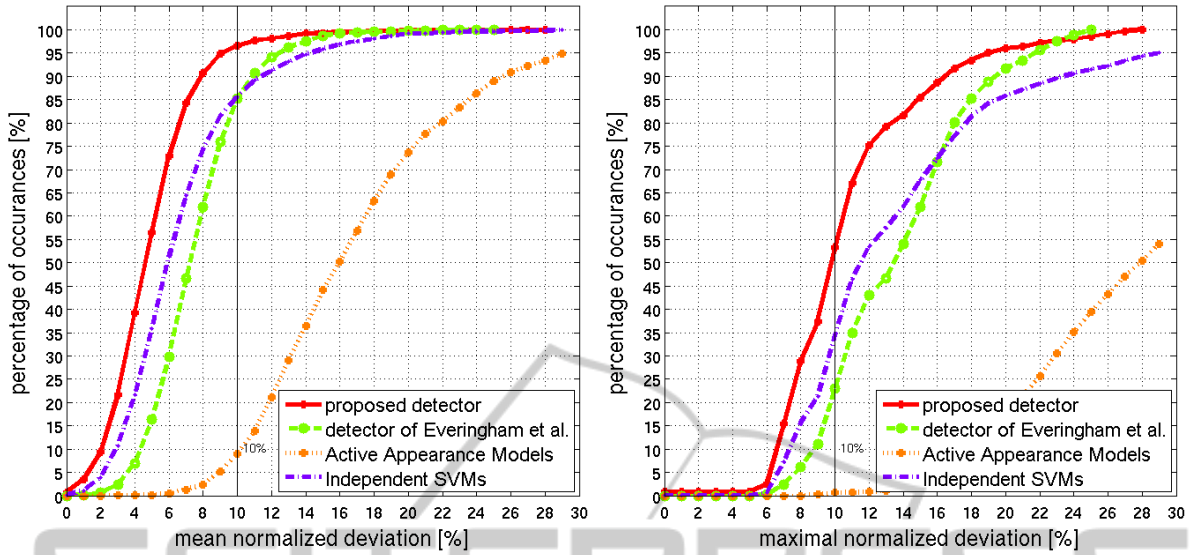



Figure 5: Cumulative histograms for the mean and the maximal normalized deviation shown for all competing detectors.

to the accuracy measure. Surprisingly, the independently trained SVM detector is comparable with the DPM based detector of (Everingham et al., 2008). The far worst results were obtained for the AAM based detector which can be attributed to a relatively low resolution of the input images.

In Figure 5 we show the cumulative histograms of the mean and maximal normalized deviation. Table 3 shows the percentage of examples from the test part of the LFW database with the mean/maximal normalized deviation less or equal to 10% (this corresponds to the line at 10% of x-axis taken from Figure 5. It is seen that the proposed detector estimates around 97% of images with the mean normalized deviation less than 10%. This results is far better than was achieved for all other competing methods. In Figure 6, we show examples of images with the mean normalized deviation equal to 10% for better understanding of these statistics.

We have also measured the average time required by the proposed detector to process a single image. The measurements were done on a notebook with Intel Core 2 Duo T9300 2.50 GHz. The average detection time is 8 ms per image.

## 4.5 Comparison of BMRM and SGD

In this section, we compare performance of the BMRM and the SGD algorithm on the problem emerging when learning the proposed detector. The task of the solvers is to minimize the problem stated in (7). Besides the value of the objective function $F(\mathbf{w})$ of the task (7) we also measured the validation risk

Table 2: Average mean normalized deviation and the average maximal normalized deviation computed on the test part of the LFW database.

| | $R_{\text{TST}}$ | $R^{\max}_{\text{TST}}$ |
|---|---|---|
| AAM | 17.6042 | 31.2715 |
| Independent SVMs | 7.1970 | 18.3601 |
| Everingham et al. | 7.9975 | 15.9451 |
| **proposed detector** | **5.4606** | **12.4080** |

Table 3: The percentage of images from the test part of the LFW database where the mean/maximal normalized deviation of the estimated landmark positions was less or equal to 10%.

| | Mean | Maximal |
|---|---|---|
| AAM | 8.98% | 0.62% |
| Everingham et al. | 85.28% | 22.93% |
| binary SVM | 85.66% | 34.50% |
| **proposed detector** | **96.59%** | **53.23%** |

$R_{\text{VAL}}(\mathbf{w})$ being another important criterion characterizing convergence of the learning algorithm.

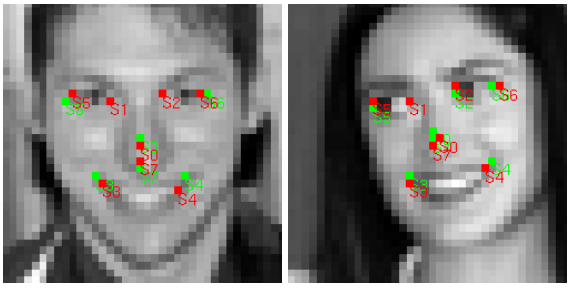
To make the iterations of both algorithms comparable, we define one iteration of the SGD as a sequence of single update steps equal to the number of training examples. This makes the computational time of both solvers approximately proportional to the number of iterations. The optimal value of parameter $t_0$ for SGD was selected to minimize the objective function $F(\mathbf{w})$ computed on 10% of the training examples after one pass of the SGD algorithm thorough the data. The parameter $t_0$ have to be tuned for each value of $\lambda$ separately. We fixed the total number of iterations of the SGD algorithm to 50.

Table 4: Comparison of the BMRM and the SGD. We show the value of primal objective function $F(\mathbf{w})$ and validation risk $R_{\mathrm{VAL}}$ for the 50th iteration (assuming termination of SGD after this iteration) as well as for the number of iterations needed by the BMRM algorithm to find the $\varepsilon$-precise solution.

| | $\lambda$ | $\lambda = 1$ | | $\lambda = 0.1$ | | $\lambda = 0.01$ | | $\lambda = 0.001$ | |
|---|---|---|---|---|---|---|---|---|---|
| | # of iterations | 50 | 106 | 50 | 201 | 50 | 462 | 50 | 1200 |
| BMRM | $F(\mathbf{w})$ | 77.48 | 62.19 | 45.13 | 29.68 | 35.33 | 14.62 | 34.35 | 7.459 |
| | $R_{\mathrm{VAL}}(\mathbf{w})$ | 23.24 | 10.48 | 9.054 | 6.067 | 9.054 | 5.475 | 9.054 | 5.876 |
| SGD | $F(\mathbf{w})$ | 50.88 | 50.44 | 20.62 | 20.52 | 13.72 | 10.80 | 12.86 | 6.309 |
| | $R_{\mathrm{VAL}}(\mathbf{w})$ | 9.719 | 9.627 | 6.156 | 6.142 | 5.577 | 5.496 | 5.544 | 5.818 |

We run both solver on the problem (7) with the parameters $\lambda \in \{0.001, 0.01, 0.1, 1\}$ recording both $F(\mathbf{w})$ and $R_{\mathrm{VAL}}(\mathbf{w})$. Results of the experiment are summarized in Table 4.

It can be seen that the SGD converges quickly at the beginning and it stalls as it approaches the minimum of the objective $F$. Similarly for the validation risk. The optimal value of $\lambda$ minimizing the validation error was 0.01 for both SGD and BMRM. The test errors computed for the optimal $\lambda$ were $R_{\mathrm{TST}} = 5.44$ for the SGD and $R_{\mathrm{TST}} = 5.54$ for the BMRM, i.e., the difference is not negligible. The results for the SGD could be improved by using more than 50 iterations, however, in that case both algorithms would require the comparable time. Moreover, without the reference solution provided by the BMRM one would not know how to set the optimal number of iterations for the SGD. We conclude that for the tested problem the BMRM produced more accurate solution, but the SGD algorithm was significantly faster. This suggests that the SGD is useful in the cases when using the precise but slower BMRM algorithm is prohibited. In the opposite case the BMRM algorithm returning a solution with the guaranteed optimality certificate is preferable.



(a) (b)

Figure 6: Sample images where the estimated landmark positions have the mean normalized deviation equal to 10%. The green and red points denote the manually annotated and estimated landmarks, respectively.

## 5 OPEN-SOURCE LIBRARY AND LFW ANNOTATION

We provide an open-source library which implements the proposed DPM detector as well as the BMRM algorithm for learning its parameters from annotated examples. The detector itself is implemented in C but we also provide a MEX interface to MATLAB. The library comes with several example applications written in C, e.g. running the detector on still images or on a video stream from a web camera. The BMRM algorithm is implemented in MATLAB up to time-critical operations which are in C. The library is licensed under the GNU/GPL version 3 and it was tested under GNU/Linux and Windows platform.

In addition, we provide a manual annotation of the LFW database for non-commercial use. The following set of landmarks is annotated for each face: the centers of both eyes, canthi for both eyes, the tip of the nose, the center of the mouth and the corners of mouth, i.e. 10 annotated landmarks in total for each image. The library and the annotation can be downloaded from: http://cmp.felk.cvut.cz/∼uricamic/flandmark

## 6 CONCLUSIONS

In this paper, we have formulated the detection of facial landmarks as an instance of the structured output classification problem. Our structured output classifier is based on the DPM and its parameters can be learned from examples by the SO-SVM algorithm. In contrast to the previous works, the learning objective is directly related to the accuracy of the resulting detector. Experiments on the LFW database show that the proposed detector consistently outperforms a baseline independently trained SVM detector and two public domain detectors based on the AAM and DPM.

We provide an open-source implementation of the proposed detector and the manual annotation of facial landmark for the LFW

database. Both can be downloaded from: http://cmp.felk.cvut.cz/∼uricamic/flandmark

## ACKNOWLEDGEMENTS

The first two authors were supported by EC project FP7-ICT-247525 HUMAVIPS. The second author was also supported by EC project PERG04-GA-2008-239455 SEMISOL. The last authors was supported by the Czech Ministry of Education project 1M0567.

## REFERENCES

Beumer, G., Tao, Q., Bazen, A., and Veldhuis, R. (2006). A landmark paper in face recognition. In *7th International Conference on Automatic Face and Gesture Recognition (FGR-2006)*. IEEE Computer Society Press.

Beumer, G. and Veldhuis, R. (2005). On the accuracy of EERs in face recognition and the importance of reliable registration. In *5th IEEE Benelux Signal Processing Symposium (SPS-2005)*, pages 85–88. IEEE Benelux Signal Processing Chapter.

Bordes, A., Bottou, L., and Gallinari, P. (2009). Sgdqn: Careful quasi-newton stochastic gradient descent. *Journal of Machine Learning Research*, 10:1737–1754.

Cootes, T., Edwards, G. J., and Taylor, C. J. (2001). Active appearance models. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(6):681–685.

Crandall, D., Felzenszwalb, P., and Huttenlocher, D. (2005). Spatial priors for part-based recognition using statistical models. In *In CVPR*, pages 10–17.

Cristinacce, D. and Cootes, T. (2003). Facial feature detection using adaboost with shape constraints. In *14th Proceedings British Machine Vision Conference (BMVC-2003)*, pages 231–240.

Cristinacce, D., Cootes, T., and Scott, I. (2004). A multistage approach to facial feature detection. In *15th British Machine Vision Conference (BMVC-2004)*, pages 277–286.

Erukhimov, V. and Lee, K. (2008). A bottom-up framework for robust facial feature detection. In *8th IEEE International Conference on Automatic Face and Gesture Recognition (FG2008)*, pages 1–6.

Everingham, M., Sivic, J., and Zisserman, A. (2006). "Hello! My name is... Buffy" – automatic naming of characters in TV video. In *Proceedings of the British Machine Vision Conference*.

Everingham, M., Sivic, J., and Zisserman, A. (2008). Willow project, automatic naming of characters in tv video. MATLAB implementation, www: http://www.robots.ox.ac.uk/∼vgg/research/nface/index.html.

Everingham, M., Sivic, J., and Zisserman, A. (2009). Taking the bite out of automatic naming of characters in TV video. *Image and Vision Computing*, 27(5).

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., and Ramanan, D. (2009). Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(1).

Felzenszwalb, P. F. and Huttenlocher, D. P. (2005). Pictorial structures for object recognition. *Internatinal Journal of Computer Vision*, 61:55–79.

Fischler, M. A. and Elschlager, R. A. (1973). The representation and matching of pictorial structures. *IEEE Transactions on Computers*, C-22(1):67–92.

Franc, V. and Sonnenburg, S. (2010). Libocas — library implementing ocas solver for training linear svm classifiers from large-scale data. www: http://cmp.felk.cvut.cz/ xfrancv/ocas/html/index.html.

Heikkilä, M., Pietikäinen, M., and Schmid, C. (2009). Description of interest regions with local binary patterns. *Pattern Recognition*, 42(3):425–436.

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

Kroon, D.-J. (2010). Active shape model (ASM) and active appearance model (AAM). MATLAB Central, www: http://www.mathworks.com/matlabcentral/fileexchange/26706-active-shape-model-asm-and-active-appearance-model-aam.

Nordstrøm, M. M., Larsen, M., Sierakowski, J., and Stegmann, M. B. (2004). The IMM face database - an annotated dataset of 240 face images. Technical report, Informatics and Mathematical Modelling, Technical University of Denmark, DTU.

Riopka, T. and Boult, T. (2003). The eyes have it. In *Proceedings of ACM SIGMM Multimedia Biometrics Methods and Applications Workshop*, pages 9–16.

Sivic, J., Everingham, M., and Zisserman, A. (2009). "Who are you?" – learning person specific classifiers from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Teo, C. H., Vishwanthan, S., Smola, A. J., and Le, Q. V. (2010). Bundle methods for regularized risk minimization. *J. Mach. Learn. Res.*, 11:311–365.

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

Viola, P. and Jones, M. (2004). Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154.

Wu, J. and Trivedi, M. (2005). Robust facial landmark detection for intelligent vehicle system. In *IEEE International Workshop on Analysis and Modeling of Faces and Gestures*.