Random Subspace Supervised Descent Method for Computer Vision Problems

Heng Yang, Xuhui Jia, Ioannis Patras, Kwok-Ping Chan

***Abstract*—Supervised Descent Method (SDM) [9] has shown competitive performance in solving non-linear least squares problems in computer vision and gives state of the art results for the problem of face alignment. However, when SDM learns the generic descent maps, it is very difficult to avoid over-fitting on a set of training data, due to the high dimensionality of the input features. In this paper we propose a Random Subspace SDM (RSSDM) that maintains the high accuracy on training data and improves the generalization accuracy. Instead of using all the features for descent learning at each iteration, we randomly select sub-sets of the features and learn an ensemble of descent maps in the subspaces, one in each sub-set. Then, we average the ensemble of descents to calculate the update of the iteration. We test the proposed methods on two representative problems, namely, 3D pose estimation and face alignment and show that RSSDM consistently outperforms SDM in both tasks in terms of accuracy. RSSDM also holds several useful generalization properties: e.g. it is more effective when the number of training samples is small and less sensitive to the changes of the strength of the regularization.**

***Index Terms*—Supervised descent method, face alignment, 3D pose estimation, random subspace.**

I. Introduction

Newton's gradient descent method has been successfully applied in many non-linear optimization problems. However, when it is applied to computer vision problems, there are many drawbacks of this second order optimization scheme. For example, 1) some popular features like the Histogram of Oriented Gradients (**HOG**) [3] are not twice differentiable; 2) computation of the Jacobians and Hessians is very expensive. To tackle such issues, Xiong and De la Torre [9], [8] proposed a Supervised Descent Method (**SDM**). Similar to Newton's method, given an initial estimate of the state of an object $\mathrm{x}_0 \in \Re^{p\times 1}$ (e.g. this can be a p dimensional 3D pose vector of an object, or a 2D shape vector representing the locations of facial landmarks in an image), SDM creates a sequence of descent maps $\mathbf{R}_0, \cdots, \mathbf{R}_k, \cdots$. Each update step is represented as:

$$\mathrm{x}_{k+1} = \mathrm{x}_k - \mathbf{R}_k(\mathrm{h}(\mathrm{x}_k) - \mathrm{h}(\mathrm{x}_*)) \qquad (1)$$

where $\mathrm{h} : \Re^n \to \Re^m$, is a transformation that varies according to different applications. It can be regarded as a generalized feature extraction term. For instance, in face alignment case, $\mathrm{h}(\mathrm{x})$ represents the HOG values computed in the local patches extracted from the landmarks with shape x. In 3D pose estimation case, $\mathrm{h}(\mathrm{x})$ is the image projection of the 3D model points. x_* represents an optimal solution. In this way, the learned sequence $\{\mathbf{R}_k\}$ moves the initial shape vector x_0 (average face shape or 3D pose) towards the optimal solution x_*. The key contribution of SDM is the supervised learning of $\{\mathbf{R}_k\}$ by minimizing the L_2 norm of the shape or 3D pose difference. It is based on a large number of training samples generated by Monte-Carlo sampling methodology. In the proposed method, $\mathbf{R}_k$ is a linear regressor. We note that for application like face alignment, since $\mathrm{x}_*)$ is unknown at test time, $\mathrm{h}(\mathrm{x}_*)$ is simplified as an additional bias term in the linear regression function during both training and testing. The SDM has shown very good performances in several important computer vision problem such as 3D pose estimation and template tracking. It is regarded as the benchmark approach for face alignment, which is a crucial step for face recognition, face animation and facial expression recognition.

However, when developing the SDM model in practice, two main problems arise:

- In order to learn an optimal $\mathbf{R}_k$, at least m training samples are usually required, with m the dimensionality of the feature space. Otherwise, the system is underdetermined. m is usually very big, for example, in face alignment application of [8], $m = 66 \times 128$ (66 is the number of facial landmarks and 128, HOG feature length). Moreover, the closed-form solution of such equations requires the inversion of matrix of size $m \times m$, which is computational expensive.
- The linear function, mapping $\Re^m \to \Re^1$, is very likely to over-fit the data during the training time. Regularization is required therefore a free parameter needs to be tuned empirically. However, when both the number of samples and the feature space are large, a single linear regression struggles to avoid over-fitting a set of training data while maintaining good performance.

In this paper, we propose a Random Subspace SDM (**RSSDM**) to overcome the drawbacks mentioned above and improve the generality, inspired by the *Stochastic Discrimination* theory [6]. At each iteration, instead of learning one linear regression, we learn several of them, each of which is based on randomly selecting a small number of dimensions from the feature space, e.g. a Random Subspace. Then, we use such an ensemble of linear regressors to represent the descent map. We test the proposed method in two representative application cases of the SDM, i.e., face alignment and 3D pose estimation to demonstrate the benefits of our approach. More specifically, our method (RSSDM): 1) can naturally handle the

H. Yang and I. Patras are with the Department of Electronic Engineering and Computer Science, Queen Mary University of London, UK. email: {heng.yang, i.patras}@qmul.ac.uk.

Xuhui Jia and Kwok-Ping Chan are with the University of HongKong. Email: xhjia,kpchan@cs.hku.hk.

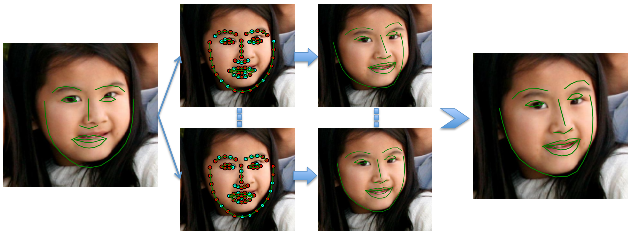

Fig. 1: RSSDM for face alignment. The image on the left shows the current pose. Then several subspaces are randomly generated, of which the cyan landmarks are selected and the red are not selected (Best viewed by zooming in). The update of one iteration is the average of outputs from several weak regressors.

under-determined issue by transforming full feature space into subspaces and shows significantly better performance when training samples are limited ; 2) shows great advantages in dealing with over-fitting and is more robust to regularization parameter changes; 3) can achieve monotonic increase in generalization accuracy w.r.t. SDM and obtain performance superior or close to other recent methods.

II. Random Subspace SDM

In the section, we first present the Random Subspace SDM for face alignment and then for 3D pose estimation. The main difference of those two applications is that, for face alignment, $\mathrm{y} = \mathrm{h}(\mathrm{x}_*)$ (i.e., the HOG features extracted from the optimal locations of facial landmarks) is unknown while for 3D pose estimation $\mathrm{y} = \mathrm{h}(\mathrm{x}_*)$ (i.e., the image projection under the optimal 3D pose) is known.

A. Random Subspace SDM for Face Alignment

Similar to the setting of other face alignment models, at training time, a set of N images $\mathcal{I} = \{I_i\}_{i=1}^N$ are available, along with their ground truth locations of facial landmarks $\mathrm{X} = \{\mathrm{x}^i_*\}$. Thus $\mathrm{x} \in \Re^{2p \times 1}$, with p the number of facial landmarks. In what follows we refer to x as the *shape* of a face. Similar to most of the face alignment models, our method also assumes that the face detection is available both in the training and in test images. We represent the face bounding box from the face detector as $b^i = (b^i_c, b^i_w, b^i_h)$, with $b^i_c \in \Re^2$ the face center, b^i_w the width and b^i_h the height. Then the location of the j-th landmark vector $\mathrm{x}^{i,j}$, containing the x and y coordinates, can be translated by the box center and scaled by the box size, which we will refer to as normalized by b^i:

$$\mathcal{N}(\mathrm{x}^{i,j}; b^i) = \begin{pmatrix} \frac{1}{b^i_w} & 0 \\ 0 & \frac{1}{b^i_h} \end{pmatrix} (\mathrm{x}^{i,j} - b^i_c) \tag{2}$$

Since the face box provides the scale information, the image is transformed as to ensure the face box is at a canonical size (width and height), which is denoted by $(\bar{b}_w, \bar{b}_h)$. The scale factors are (s^i_w, s^i_h) with $s^i_w = \frac{\bar{b}_w}{b^i_w}$, $s^i_h = \frac{\bar{b}_h}{b^i_h}$. The initial shape estimate is given by centering the mean face at the canonical face box, that is denoted by x_0. In the rest of the paper, we assume the shape vectors and the images are transformed by the face boxes. We also generated 10 samples by perturbing the face box for each training image using Monte Carlo methodology as described in [8]. This augments the training samples by a factor of 10. We will treat each of them as a unique sample. Then for the i-th sample, the desired update (error vector) is $\Delta \mathrm{x}^i_0 = \mathrm{x}^i_* - \mathrm{x}^i_0$. HOG features around each landmark under the current shape are extracted $\widetilde{\phi}^i_0 = h(I^i, x^i_0)$. Since x_* is not available for this problem we added a bias term to the feature vector for linear regression so that the feature vector becomes $\phi^i_0 = [(\widetilde{\phi}^i_0)^T, 1]^T$. Thus we seek for $\mathbf{R}_0$ that minimizes:

$$\arg\min_{\mathbf{R}_0} \sum_i ||\Delta \mathrm{x}^i_0 - \mathbf{R}_0 \phi^i_0||^2 \tag{3}$$

The above least squares problem can be solved in closed-form given sufficient samples (equations). Then by applying the learned regressor $\mathbf{R}_{k-1}$, we can update the current shape x^i_k by adding the update. The new optimal update becomes $\Delta \mathrm{x}^i_k = \mathrm{x}^i_* - \mathrm{x}^i_k$ and the new feature vector is ϕ^i_k. A new regressor $\mathbf{R}_k$ can be learned by minimizing:

$$\arg\min_{\mathbf{R}_k} \sum_i ||\Delta \mathrm{x}^i_k - \mathbf{R}_k \phi^i_k||^2. \tag{4}$$

This is the training process of SDM described in [8]. In order to avoid over-fitting, we also introduce a regularization term for SDM and the optimization becomes:

$$\arg\min_{\mathbf{R}_k} \sum_i ||\Delta \mathrm{x}^i_k - \mathbf{R}_k \phi^i_k||^2 + \lambda ||\mathbf{R}_k||^2_F. \tag{5}$$

This formulation requires tuning the λ very well therefore cross validation is usually applied to search for the optimal λ. However when the size of training samples is large, which is to guarantee closed-form solution, selecting a proper λ is intractable. Encouraged by the success of Random Subspace in tree construction [4], which also faces the over-fitting issue, we adapt it for SDM. More specifically, instead of using the whole feature space, we select several random subspaces and train an ensemble of regressors in subspaces. For this face alignment case, we still keep the feature structure extracted from one landmark location. As shown in Fig. 1, from the set of landmarks $J = \{j\}^p_{j=1}$, we select several subsets, $\{J_t\}^T_{t=1}$, with $J_t \subset J$. We denote the features exacted from the landmarks in the t-th subset as $\phi^{i,t}_k$, $\phi^{i,t}_k \subset \phi^i_k$. We then train T regressors, one on each subset, using the corresponding features. We then optimize the following function:

$$\arg\min_{\mathbf{R}^t_k} \sum_i ||\Delta \mathrm{x}^i_k - \mathbf{R}^t_k \phi^{i,t}_k||^2 + \lambda^t ||\mathbf{R}^t_k||^2_F. \tag{6}$$

for each of $\mathbf{R}^t_k, t = 1, ..., T$, regressors. We then simply average the outputs of such an ensemble of regressors to update the current shape. That is

$$\mathrm{x}^i_{k+1} = \mathrm{x}^i_k - \sum_{t=1}^{T} \mathbf{R}^t_k \phi^{i,t}_k. \tag{7}$$

A recursive procedure similar to the SDM is applied to the cascade framework when the shape of each sample is updated until the final iteration is applied. During testing time, since we have normalized the image using Eq. 2, we apply the inverse of of the normalization function to transform the final shape vector and obtain the alignment result. Assuming that the shape estimation after applying the final iteration is x_K^i, then the final shape estimation is:

$$\hat{\mathrm{x}}^i = \mathcal{N}^{-1}(\mathrm{x}_K^i; b^i). \quad (8)$$

B. Random Subspace SDM for 3D Pose Estimation

In this section we present how we apply the random subspace SDM to another computer vision problem, 3D pose estimation. This problem can be described as follows. Given the 3D model of an object represented as 3D points $\mathrm{M} \in \mathfrak{R}^{3\times p}$, its image projection $\mathrm{U} \in \mathfrak{R}^{2\times p}$ and the intrinsic camera parameters $\mathrm{K} \in \mathfrak{R}^{3\times 3}$, the goal is to estimate the 3D object pose, consisting of a rotation vector ($\theta \in \mathfrak{R}^{3\times 1}$) and a translation vector $tr \in \mathfrak{R}^{3\times 1}$. To be consistent, we denote the pose vector by $\mathrm{x} = [\theta; tr]$. Then the objective function becomes $||\mathrm{h}(\mathrm{x}, \mathrm{M}) - U||_F$, with a known K. Given a set of poses $\{\mathrm{x}_*^i\}$ and the image projections U^i, the SDM optimization is defined as:

$$\arg\min_{\mathbf{R}_k} \sum_i ||\mathrm{x}_*^i - \mathrm{x}_k^i + \mathbf{R}_k(\mathrm{h}(\mathrm{x}_k^i, \mathrm{M}) - \mathrm{U}^i)||_2^2. \quad (9)$$

Similar to the RSSDM for face alignment, we propose to use an ensemble of regressors in subspaces at each iteration. We denote by $\phi_k^i = \mathrm{h}(\mathrm{x}_k^i, \mathrm{M})$ the features extracted based on the current pose x_k^i, and $\phi_k^{i,t}$ the feature in subspace t. The corresponding image projection is $\mathrm{U}^{i,t}$. Similar to Eq. 6, the optimization of the regressor in subspace t is as follows,

$$\arg\min_{\mathbf{R}_k^t} \sum_i ||\mathrm{x}_*^i - \mathrm{x}_k^i + \mathbf{R}_k^t(\phi_k^{i,t} - \mathrm{U}^{i,t})||_2^2 + \lambda^t ||\mathbf{R}_k^t||_F^2. \quad (10)$$

The update of the pose is calculated in a way similar to Eq. 7. At testing time, with a sequence of descent maps, the RSSDM always starts at the mean pose x_0, similar to the SDM method, and converges to the optimal solutions.

III. Experiments

A. Face Alignment

We carry out the experiment of face alignment on the benchmark 300W [7] database. We reimplement the baseline SDM following the description in [9] for fair comparison. We train the baseline SDM and our proposed RSSDM using the training images in HELEN subset from 300W. In order to train an optimal model for both SDM and RSSDM, at each iteration we search for the optimal penalty parameter in a big space by 10-fold cross validation. We terminate the cascade when the error on the training set is lower than a threshold. In this way, we get very close training error for SDM and RSSDM.

We do a grid search for setting the parameters of RSSDM. More specifically, we set the number of subspaces in the range of $N_{SP} = [2:2:10]$ and the subspace feature dimensionality in the range of $D_{SP} = \frac{D}{[2:1:6]}$, where D is the dimensionality of the original feature space. Each combination of them is evaluated separately and we report their results in Fig. 2. When the number of subspaces is very low, decreasing the subspace dimension (using less features) will lead to larger error. When the number of subspaces is at a moderate number (6 or 8), the optimal subspace dimension lies in the middle. We select the second best combination of ($N_{SP} = 6$ and $D_{SP} = \frac{1}{3}$) in our following experiments as it has similar run-time cost as the original SDM (35FPS (SDM) vs. 38FPS (RSSDM) on the same machine) while keeping good performance.

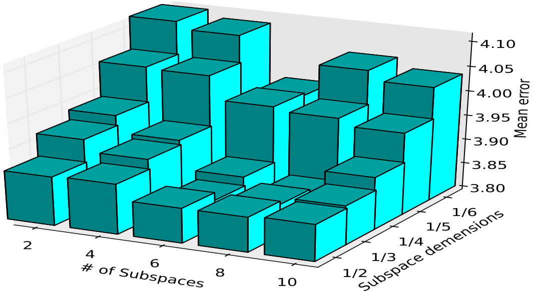

Fig. 2: RSSDM performance with various number of subspaces and subspace dimensions.

1) RSSDM vs. SDM : Then we test the model on the test images on both the Easy-set (test images from LFPW and HELEN) and Challenging-set (iBug images). As the results shown in Fig. 3, RSSDM consistently performs better than SDM on both the Easy set and the Challenging set. Since the performance on the Easy set is near saturation, with the detection rate close to 100% at the error rate of 0.15, the improvement of RSSDM over SDM is small. The improvement on the Challenging set is larger, with 4% improvement at error rate of 0.1. The overall improvement is not huge but significant. The statistical hypothesis (the same mean error) test is rejected with high confidence (p <0.0001). Moreover, as we will show in the following, the proposed RSSDM scheme has benefits in certain circumstances, while still keeping monotonically increasing performance in accuracy w.r.t. SDM.

2) Sensitivity to number of Monte-Carlo permutations: In this section, we compare the performance of RSSDM and SDM when the permutation number changes. As stated in [9], the generic DM only exists within a local neighbourhood of the optimal parameters. Therefore in the training process, the number of Monte-Carlo permutations affect the results significantly. In this section, we evaluate the sensitivity of our RSSDM and SDM to the Monte-Carlo number. We set the system parameters including the regularization parameters and the number of iterations of both methods to the optimal ones learned from above section. Then we decrease the permutation number from 9 to 1 with step size 2 and calculate their performance on the Easy and Challenging test sets respectively. The result is shown in Fig. 4. As expected, the error of both SDM and RSSDM increases while the number of Monte-Carlo permutations decreases. However, the impact on RSSDM is less. On the easy set, the mean error increases from 2.74% to 2.85% for RSSDM while that of SDM increases from 2.83% to 3.26%. On the Challenging set, the mean error of RSSDM increases from 8.55% to 9.03% while that of SDM increases from 8.89% to 10.22%. Based on this observation, we can make the conclusion that, the proposed RSSDM is less sensitive to Monte-Carlo permutation reduction. Another conclusion we can draw from this experiment is that, RSSDM is able to obtain better performance when the training samples

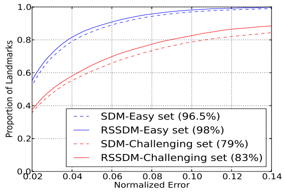

Fig. 3: SDM vs. RSSDM

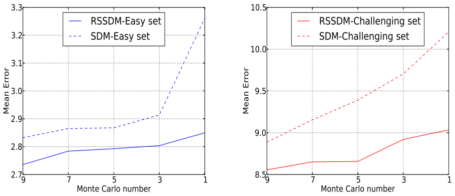

Fig. 4: Monte-Carlo numbers.

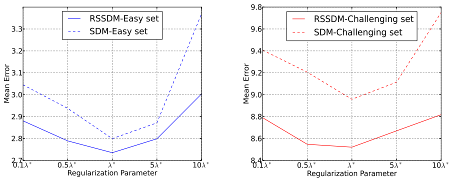

Fig. 5: Sensitivity to λ

are limited. RSSDM with 3 Monte-Carlo permutations can achieve similar performance to SDM with 9 Monte-Carlo permutations. This is a very useful property under the circumstance when it is intractable to generate a large number of Monte-Carlo samples.

3) Sensitivity to λ: In this section, we measure the sensitivity of RSSDM and SDM to the regularization parameter λ. In the previous discussion, we have obtained the optimal λ at each iteration. Assuming that the optimized λ is λ^*, we retrain the models using λ with the following values $[0.1\lambda^*, 0.5\lambda^*, \lambda^*, 5\lambda^*, 10\lambda^*]$ and record their results. As can be seen in Fig. 5, when the regularization parameter shift from the optimal one, the error for both RSSDM and SDM increases. However, RSSDM shows better performances in terms of robustness to such changes. For instance, on the easy set, when λ changes from λ^* to 10 times larger, the mean error of SDM increases nearly 0.6 while that of RSSDM increases only 0.25. On the Challenging set, the error increase of SDM is 0.8 while that of RSSDM is only 0.3. This can be explained by the ensemble strategy of the RSSDM method, of which in each iteration, the update is an average of the outputs from several weak regressors.

4) Comparison to state of the art: Face alignment is a very active research topic and many techniques have been proposed [10], [13], [11], [5]. Due to the limited space, we only compare the methods that are related to SDM, including the public available code of SDM (SDM-A), our implementation of SDM (SDM-B), the Incremental Face Alignment (IFA) model in [2] and CFAN [12]. The IFA is a variant of SDM that can be trained in a parallel way and also trained on the 300W dataset using HOG features. CFAN also follows the SDM scheme and is trained on 300W but uses features learned from auto-encoder networks. The localization error of the inner 49 facial landmarks are recorded, as SDM-A does. In order to be consistent to [8], [2], the error is normalized by the inter-ocular distance instead of the face size in this experiment. Our implementation of SDM (SDM-B) performs on par with the publicly available model (SDM-A), which validates the implementation process. The proposed RSSDM outperforms the two versions of SDM as well as IFA. Though on the easy set, by using more complicated learned features, CFAN performs slightly better than RSSDM, on the difficult set, RSSDM has better performance.

TABLE I: 300-W dataset (49 landmarks).

Method	Full-set	Easy-set	Challenging-set
CFAN [12]	7.24	**4.85**	17.04
IFA [2]	8.30	5.48	19.88
SDM-A [8]	7.06	5.56	13.22
SDM-B	6.86	5.45	12.66
RSSDM	**6.17**	4.95	**11.20**

B. 3D Pose Estimation

In this section we evaluate the performance of RSSDM on another computer vision problem, 3D pose estimation. As we discussed before, our method is proposed for the situation that the feature space is much bigger than the output space. Thus we use a human body 3D pose estimation [1] to demonstrate the performance. The body consists of 996 3D key points thus its image projection contains 996×2 dimensions of features. We follow the experimental setting as [9]. More specificity, the virtual camera is at the origin of the coordinate system and the intrinsic parameters are: focal length $f_x = f_y = 1000$ pixels, principle point $[u_0, v_0] = [500, 500]$. The object is placed at $[0, 0, 2000]$, and perturbed with different 3D poses. Three rotation angles are uniformly sampled from -30^o to 30^o with increments of 10^o in training and 7^o in testing. Three translation values are uniformly sampled from -400mm to 400mm with increments of 200mm in training and 170mm in testing. For each combination we get one training sample. We also add white noise ($\sigma^2 = 4$) on the projected points and normalize the projection by the focal length and the principle point of the camera. We also do a grid search for both SDM and RSSDM for the optimal parameters by cross validation on the training set. The result is shown in Table II. As a re-implementation, our result of SDM is slightly different from [9]. As can be seen in the table, both SDM and RSSDM outperform the POSIT algorithm with a large margin. The RSSDM further improves the accuracy over SDM, which validates the efficacy of our proposed method in 3D pose estimation application.

TABLE II: Rotation (^{o}) and translation (in mm) errors.

Method	θ_x	θ_y	θ_z	tr_x	tr_y	tr_z
POSIT	0.6±0.6	6.3±5.3	2.1±1.6	22.3±14.8	14.9±11.2	41.1±38.0
SDM	0.07±0.05	0.25±0.15	0.2±0.11	3.7±3.0	4.1±3.6	6.5±5.3
RSSDM	0.06±0.04	0.22±0.13	0.15±0.09	3.4± 3.1	3.7±3.2	5.2±4.3

IV. Conclusion

In this paper, we proposed a simple yet effective Random Subspace Supervised Descent Method (RSSDM). We compare RSSDM to SDM on Face Alignment and 3D pose estimation and obtain better performance in estimation accuracy. It also holds several other interesting properties, i.e., RSSDM is significantly more effective than SDM when the Monte-Carlo number is small and less sensitive to the regularization term. We believe these properties are important in designing a real system. As an initialization dependent method, similar to SDM, RSSDM is still sensitive to unreliable initialization. Thus, we will investigate a reliable way of initialization in future research.

References

[1] Nancy Body Model, available at www.robots.ox.ac.uk/wmayol/3D/nancymatlab.html.

[2] A. Asthana, S. Zafeiriou, S. Cheng, and M. Pantic. Incremental face alignment in the wild. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2014.

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. Computer Vision and Pattern Recognition*, 2005.

[4] T. K. Ho. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

[5] X. Jia, H. Yang, A. Lin, K.-P. Chan, and I. Patras. Structured semi-supervised forest for facial landmarks localization with face mask reasoning. 2014.

[6] E. M. Kleinberg. Stochastic discrimination. *Annals of Mathematics and Artificial Intelligence*, 1:207–239, 1990.

[7] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic. 300 faces in-the-wild challenge: The first facial landmark localization challenge. In *Proc. IEEE Int'l Conf. Computer Vision*, 2013.

[8] X. Xiong and F. De la Torre. Supervised descent method and its applications to face alignment. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2013.

[9] X. Xiong and F. De la Torre. Supervised descent method for solving nonlinear least squares problems in computer vision. *arXiv:1405.0601*, 2014.

[10] H. Yang and I. Patras. Sieving regression forests votes for facial feature detection in the wild. In *Proc. Int'l Conf. Computer Vision*, 2013.

[11] H. Yang, C. Zou, and I. Patras. Face sketch landmarks localization in the wild. *IEEE Signal Processing Letters*, 21:1321–1325, 2014.

[12] J. Zhang, S. Shan, M. Kan, and X. Chen. Coarse-to-fine auto-encoder networks (cfan) for real-time face alignment. In *Proc. European Conf. Computer Vision*, 2014.

[13] Z. Zhang, P. Luo, C. C. Loy, and X. Tang. Facial landmark detection by deep multi-task learning. In *Proc. European Conf. Computer Vision*, pages 94–108. Springer, 2014.