Hierarchical Face Parsing via Deep Learning

Ping Luo[1,3] Xiaogang Wang[2,3] Xiaoou Tang[1,3]

[1]Department of Information Engineering, The Chinese University of Hong Kong

[2]Department of Electronic Engineering, The Chinese University of Hong Kong

[3]Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences

pluo.lhi@gmail.com xgwang@ee.cuhk.edu.hk xtang@ie.cuhk.edu.hk

Abstract

This paper investigates how to parse (segment) facial components from face images which may be partially occluded. We propose a novel face parser, which recasts segmentation of face components as a cross-modality data transformation problem, i.e.*, transforming an image patch to a label map. Specifically, a face is represented hierarchically by parts, components, and pixel-wise labels. With this representation, our approach first detects faces at both the part- and component-levels, and then computes the pixel-wise label maps (Fig.1). Our part-based and component-based detectors are generatively trained with the deep belief network (DBN), and are discriminatively tuned by logistic regression. The segmentators transform the detected face components to label maps, which are obtained by learning a highly nonlinear mapping with the deep autoencoder. The proposed hierarchical face parsing is not only robust to partial occlusions but also provide richer information for face analysis and face synthesis compared with face keypoint detection and face alignment. The effectiveness of our algorithm is shown through several tasks on* $2,239$ *images selected from three datasets (*e.g.*, LFW [12], BioID [13] and CUFSF [29]).*

1. Introduction

Explicitly parsing face images into different facial components implies analyzing the semantic constituents (*e.g.*, mouth, nose, and eyes) of human faces, and is useful for a variety of tasks, including recognition, animation, and synthesis. All these applications bring new requirements on face analysis—robustness to pose, background, and occlusions. Existing works, including both face keypoint detection and face alignment, focus on localizing a number of landmarks, which *implicitly* cover the regions of interest. The main idea of these methods is to first initialize the locations of the landmarks (*i.e.*, mean shape) by classification or regression, and then to refine them by template matching [2, 3, 17, 16, 5, 30, 14] or graphical models (*e.g.*, MRF) [23, 15]. In this work, we study the problem from a new point of view and focus on computing the pixel-wise label map of a face image as shown in Fig.1 (d). It provides richer information for further face analysis and synthesis such as 3D modeling [1] and face sketching [20, 19, 26], comparing to the results obtained by face keypoint detection and face alignment. This task is challenging and existing image segmentation approaches cannot achieve satisfactory results without human interaction. An example is shown in Fig.1(e). Inspired by the success of deep autoencoder [11], which can transform high-dimensional data into low-dimensional code and then recover the data from the code within single modality, we recast face component segmentation as a cross-modality data transformation problem, and propose an alternative deep learning strategy to directly learn a highly non-linear mapping from images to label maps.

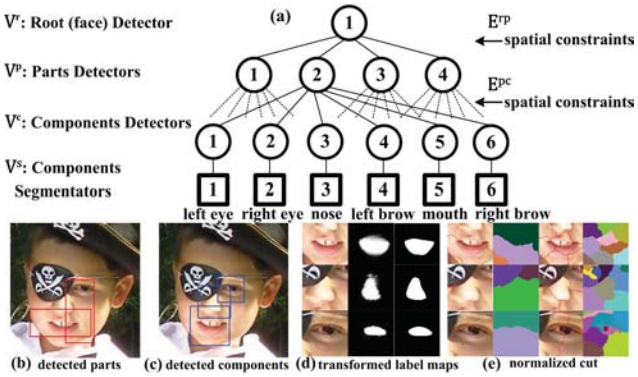

Figure 1. Hierarchical representation of face parsing (a). A face image is parsed by combining part-based face detection (b), component-based face detection (c), and component segmentation (d). There are four part-based detectors (left/right/upper/lower-half face) and six component-based detectors (left/right eye, left/right eyebrow, nose and mouth). Each component detector links to a novel component *segmentator*. (d) shows that the segmentators can transform the detected patches to label maps ($1^{st}, 2^{nd}$ columns) and we obtain the fine-label maps after hysteresis thresholding (3^{rd}). (e) is the image segmentation result (*i.e.*, groups into 2 and 10 clusters) obtained by normalized cut.

978-1-4673-1228-8/12/$31.00 ©2012 IEEE

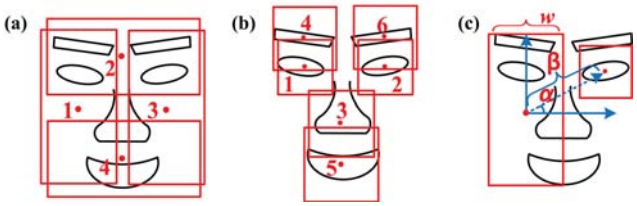

Figure 2. (a) and (b) define parts and components, which correspond to the nodes in Fig.1 (a). Red points are the positions of parts (a) or components (b). Red boxes are extracted image patches for training. (c) shows the spatial relationship (*i.e.*, orientation and position) between parts and components.

In this paper, we propose a hierarchical face parser. A face is represented hierarchically by parts, components, and pixel-wise labels (Fig.1 (a)). With this representation, our approach first detects faces at both the part- and component-levels, and then computes the pixel-wise label maps. The deformable part- and component-based detectors are incorporated by extending the conventional Pictorial model [9] in a hierarchical fashion. As shown in Fig.1 (a), each node at the upper three layers is a detector generatively pre-trained by Deep Belief Network (DBN) [10] and discriminatively tuned with logistic regression (sec.3.2). At the bottom layer, each node is associated with a component *segmentator*, which is fully trained on a dataset with patch-label map pairs by a modified deep autoencoder [11] (sec.3.3). Then, we demonstrate that with this model, a greedy search algorithm with a *data-driven* strategy is sufficient to efficiently yield good parsing results (sec.3.1). The effectiveness of hierarchical face parsing is demonstrated through the applications of face alignment and detection of facial keypoints, and it outperforms the state-of-the-art approaches.

1.1. Related Work

Recent approaches of scene parsing [22] provide an alternative view for face analysis, which is to compute the pixel-wise label maps [27]. This representation offers richer information and robustness compared with the existing face alignment and key point detection methods.

Active Shape Model (ASM) [3] is a well established and representative face alignment algorithm, and has many variants [2, 17, 30]. They heavily rely on good initialization and do not work well on images taken in unconstrained environments, where shape and appearance may vary greatly. Starting with multiple initial shapes is a natural way to overcome this problem [21, 15]. For instance, in order to be robust to noise, the Markov Shape Model [15, 14] samples many shapes by combining the local line segments and appearances as constraints. Although such methods reduce the dependence on initialization, they are computationally expensive since a large number of examples have to be drawn; otherwise, the matching may get stuck at local minimum.

To improve computational efficiency and to be robust to pose variations and background clutters, it has become popular to adopt discriminative approaches in facial analysis [23, 16, 5, 28]. For example, the BoRMaN method [23] and the Regression ASM [5] were proposed to detect facial features or components using boosting classifiers on small image patches. A component-based discriminative search algorithm [16] extended the Active Appearance Model (AAM) [2] by combining the facial-component detectors and the direction classifiers, which predict the shifting directions of the detected components.

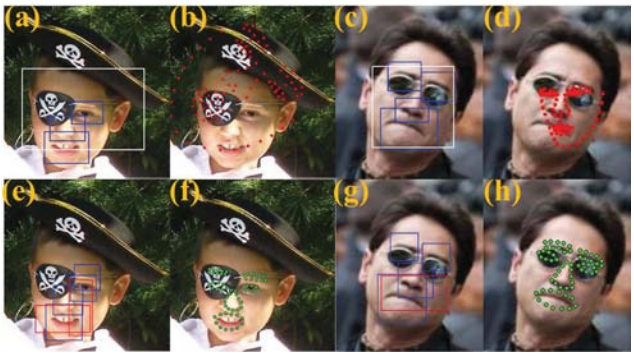

Figure 3. Compare the alignment results of [16] ((a)-(d)) and ours ((e)-(h)) when face images are partially occluded. Landmarks can be easily obtained from the label maps obtained by our approach. The white boxes indicate the initial face detection results employed in [16]. It is not accurate in (a) due to occlusion. The blues boxes indicate the results of component detection. The red boxes indicate the results of part-based detectors employed in our approach.

However, the aforementioned approaches have two drawbacks. First, it is difficult for a single detector [25] to accurately locate an occluded face for initialization. Thus, the matching process fails to converge since the initial shape is far away from the optimal solution as shown in Fig.3 (a). Our method adopts multiple hierarchical deformable part- and component-based detectors and is more robust to partial occlusions. Second, since face detection and shape matching are optimized alternately, we empirically observe that even though the face components can be well detected, shape matching may still converge to a local minima because the correlation between shape and image appearance is not well captured as shown in Fig.3 (a)(c). Our method employs DBN to establish strong correlations between images and shapes by estimating the label maps directly from the detected image patches. See Fig.1 (d) and Fig.3 (f)(h) for details.

2. A Bayesian Formulation

Our hierarchical face parsing can be formulated under a Bayesian framework, under which the detectors and segmentators can be explained as likelihoods, and spatial consistency can be explained as priors.

Let $\mathbf{I}$ be an input image and L be a set of class labels of all the detectors. Here, $L = \{\ell^r, \ell_i^p, \ell_j^c\}_{i=1..4}^{j=1..6}$ (see the upper

three layers of Fig.1 (a))[1]. Under the Bayesian Framework, our objective is to compute a solution θ that maximizes a posterior probability, $p(\theta|\mathbf{I}, L)$. Therefore,

$$\begin{aligned}\theta^* &= \arg\max p(\mathbf{I}, L|\theta)p(\theta) \\ &= \arg\max \log p(\mathbf{I}, L|\theta) + \log p(\theta).\end{aligned} \quad (1)$$

After taking "log", the objective value is equivalent to a summation of a set of scores. In other words, our problem is, given a facial image $\mathbf{I}$, to hierarchically find the most possible parsing representation $\theta = (V^r, V^p, V^c, V^s, E)$, which contains a set of nodes and edges. More specifically, $V^{r/p/c} = \{v_i^{r/p/c} = (b_i^{r/p/c}, \rho_i^{r/p/c}, \lambda_i^{r/p/c})\}_{i=1}^K$ are the root detector ($K = 1$), part detectors ($K = 4$), and component detectors ($K = 6$) respectively. $E = (E^{rp}, E^{pc})$ indicates the spatial relations among the upper three layers. Here, we denote the component segmentators as $V^s = \{v_i^s = (b_i^s, \lambda_i^s, \phi_i, \Lambda_i)\}_{i=1}^6$. In particular, a node is described by a bounding box b, a binary variable $\lambda \in \{0, 1\}$ that indicates whether this node is occluded ("*off*") or not ("*on*"), and a set of deep learning parameters ρ and ϕ[2]. Λ denotes the label map of the corresponding component.

2.1. The Scores of Spatial Consistency

Here, $\log p(\theta)$ in Eq.1 is the score of spatial consistency, which is modeled as a hierarchical pictorial structure.

$p(\theta)$ is the prior probability, measuring the spatial compatibility between a face and its parts, and also between each part and the components. Hence, we have $p(\theta) = p(E^{rp}|v^r)p(E^{pc}|V^p)$, in which $E^{rp} = \{< v^r, v_i^p > |\forall v_i^p \in V^p\}_{i=1}^4$ and $E^{pc} = \{< v_i^p, v_j^c > |\forall v_i^p \in V^p, \forall v_j^c \in V^c\}_{i=1..4}^{j=1..6}$ indicate two edge set (Fig.1 (a)). In this paper, we consider orientations and locations as two kinds of spatial constraints. Therefore, the prior $p(E^{rp}|v^r)$ can be factorized as,

$$p(E^{rp}|v^r) = \prod_{<v^r, v_i^p> \in E^{rp}} p(o(b^r, b_i^p)|\lambda^r)p(d(b^r, b_i^p)|\lambda^r). \quad (2)$$

Here, the functions $o(\cdot,\cdot)$ and $d(\cdot,\cdot)$ respectively calculate the relative angle and the normalized Euclidean distance between the centers of two bounding boxes b^r and b^p. For example, in Fig.2 (c), we illustrate the spatial constraints of the right eye related to the left-half face. We model the probabilities of these two spatial relations as Gaussian distributions. For instance, $p(o(b^r, b^p)|\lambda^r = 1) = \mathcal{N}(o(b^r, b^p); \mu_{b^r b^p}, \Sigma_{b^r b^p})$[3]. Similarly, the prior of $p(E^{pc}|V^p)$ can be factorized in the same way.

2.2. The Scores of Detectors and Segmentators

$\log p(\mathbf{I}, L|\theta)$ in Eq.1 can be explained as the sum of the scores of detectors and segmentators, and they are modeled through DBNs and deep autoencoders respectively.

Let $\mathbf{I}_b$ be the image patch occupied by the bounding box b. The likelihood probability, $p(\mathbf{I}, L|\theta)$, can be factorized into the likelihood of each node as,

$$\begin{aligned}p(\mathbf{I}, L|\theta) &= \prod_{i \in \{r, p_1, ..., p_4, c_1, ..., c_6\}} \underbrace{p(\mathbf{I}_{b^i}, \ell^i|v^i)}_{detector} \\ &\times \prod_{j=1}^{6} \underbrace{p(\mathbf{I}_{b_j^s}|v_j^s)}_{segmentator}.\end{aligned} \quad (3)$$

By applying the Bayes rule, we formulate the likelihood of each detector as $p(\mathbf{I}_b, \ell|v) = p(\mathbf{I}_b; \rho_{dbn}) \times p(\ell|\mathbf{I}_b, \rho_{dbn}; \rho_{reg})$[4], and the likelihood of each component segmentator as $p(\mathbf{I}_{b^s}|v^s) = p(\mathbf{I}_{b^s}|\Lambda^s; \phi_{ae}^s)$[5].

We model the first term of the detector's likelihood by DBN and discuss details in sec.3.2, and the second term evaluates how likely a node should be located on a certain image patch, is derived as below,

$$p(\ell|\mathbf{I}_b, b, \lambda, \rho_{dbn}; \rho_{reg}) \propto \exp\{- \parallel \ell - f(\mathbf{I}_b, \rho_{dbn}; \rho_{reg}) \parallel_1\}, \quad (4)$$

where, given an image patch, $f(\cdot,\cdot)$ is a *softmax* function that predicts its class label based on the learned parameters ρ_{dbn} and ρ_{reg}. Furthermore, the likelihood of the segmentator has the following form[6] and its parameters are learned by deep autoencoders (sec.3.3),

$$\begin{aligned}p(\mathbf{I}_{b^s}|\Lambda^s, b^s, \lambda^s; \phi_{ae}^s) \propto \\ \exp\{-\min\{g_1(\mathbf{I}_{b^s}|\Lambda^s; \phi_{ae}^s), ..., g_k(\mathbf{I}_{b^s}|\Lambda_k^s; \phi_{ae}^s)\}\}.\end{aligned} \quad (5)$$

Here, we learn k deep autoencoders to estimate the label maps of a facial component and return the one with the minimal cross-entropy error $g_k(\cdot,\cdot)$[7].

3. Hierarchical Face Parsing

Before we discuss the details, we first give an overview of our algorithm in sec.3.1. After that, we describe our methods for learning the detectors in sec.3.2 and the segmentators in sec.3.3.

[1] In our experiment (sec.3.4), $\ell^r \in \mathbb{R}^2, \forall \ell_i^p \in \mathbb{R}^5, \forall \ell_j^c \in \mathbb{R}^7$, each vector indicates class label of the node and has a *1-of-K* representation. Consider "mouth (ℓ_5^c)" as an example, the 5-th element is set to be 1 and all the other are 0, and the 7-th element denotes the class of background.

[2] $\rho = (\rho_{dbn}, \rho_{reg})$ includes the parameters for DBN and logistic regression. ϕ will be written as ϕ_{ae} in later derivation which denotes the parameters for the deep autoencoder.

[3] When a node is "*off*", $\log p(o(b^r, b^p)|\lambda^r = 0) = \epsilon$, where ϵ is a sufficiently small negative number.

[4] We drop the superscripts r/p/c here.

[5] Note that the random variables b and λ are omitted for simplicity in the derivation of the likelihoods.

[6] Eq.4 and Eq.5 are defined when $\lambda = 1$. Please refer to footnote 3 for the situation that $\lambda = 0$.

[7] $g_k(\cdot,\cdot)$ evaluates the cross-entropy error between the input image patch and the reconstructed one under the k-th deep autoencoder.

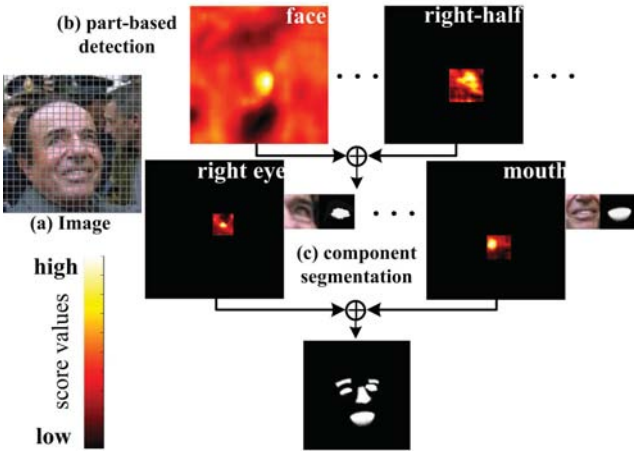

Figure 4. The parsing process. In practice, we adopt the HOG features from [6] and each testing image (a) is described by a HOG feature pyramid similar to [8]. (b) shows the scores of the "face" and the "right-half face" detectors. Note that the former one evaluates all positions while the latter one evaluates only a small portion. (c) illustrates the scores of component segmentators and the transformed label maps.

3.1. Data-driven Greedy Search Algorithm

Our data-driven greedy search algorithm can be separated into two steps: part-based face detection and component segmentation. For the first step, we assume that the nodes of root and parts are visible (*i.e.*, $\lambda = 1$), then we sequentially evaluate their detectors with the sliding window scheme. Once a node has been activated, the other nodes, guided by the data-driven information, will only search within a certain range. For instance, as shown in Fig.4 (b), the root detector tests all positions while the detector of the right-half face tests only a small portion. After running all five detectors, we combine the scores together, resulting in a good hypothesis of the face's position. Such strategy for object detection is fully evaluated in [8], where convincing results are reported. For the second step, we use the component detectors to search the components on the previously proposed location (Fig.4 (c)). If a component is activated, its corresponding segmentator is performed to output the label map. Eventually, the final score of the parsing is achieved by summing up the scores of spatial constraints (sec.2.1), detection (Eq.4), and segmentation (Eq.5). Moreover, the result will be pruned by a threshold learned on a validation set. We summarize the algorithm in Algorithm.1.

Data-driven strategy 1. To improve the search algorithm, we solve a localization problem that is to determine the angle and distance between a detected and an undetected node. For example, as illustrated in Fig.2 (c), given the location of the detected left-half face, we decide to predict the coordinates of the undetected right eye. We deal with this problem by training two regressors: the first one estimates the angle α, and the second one finds the distance β. The Support Vector Regression [4] is adopted to learn these two regressors.

Algorithm 1: Hierarchical Face Parsing

Input: an image $\mathbf{I}$ and the class label set L
Output: label maps of facial components
1) Part-based detection:
(1) evaluate face or part detectors on $\mathbf{I}$ according to sec.3.2 in a data-driven fashion
(2) hypothesize the face's or parts's position by calculating the scores of spatial constraints (Eq.2) and detection (Eq.4)
2) Component segmentation:
(1) detect the components around the location proposed by 1)
(2) if a component is detected, then estimate its label map
(3) compute the scores according to sec.2.1 and Eq.5
3) Combine the scores of 1) and 2) together, if the final score is larger than a threshold, then output the result.

Data-driven strategy 2. It is not necessary for us to enumerate all the combinations of the binary variable λ. If a node is not detectable, its λ is set to zero[8].

3.2. Learning Detectors

In this paper, we model our detectors by deep belief network (DBN), which is unsupervisedly pre-trained using layer-wise Restricted Boltzmann Machine (RBM) [10] and supervisedly fine-tuned for classification using logistic regression. Here, given image patches $\mathbf{I}_b$ as the training samples (*i.e.*, inputs in Fig.5 (a)), a DBN with K layers models the joint distribution between $\mathbf{I}_b$ and K hidden layers $\mathbf{h}^1, ..., \mathbf{h}^k$ as follows:

$$p(\mathbf{I}_b, \mathbf{h}^1, ..., \mathbf{h}^k; \rho_{dbn}) = \quad (6)$$
$$(\prod_{k=0}^{K-2} p(\mathbf{h}^k|\mathbf{h}^{k+1}; \rho_{dbn}))p(\mathbf{h}^{K-1}, \mathbf{h}^K; \rho_{dbn}),$$

where $\mathbf{I}_b = \mathbf{h}^0$, $p(\mathbf{h}^k|\mathbf{h}^{k+1}; \rho_{dbn}))$ is a visible-given-hidden conditional distribution of the RBM at level k, and $p(\mathbf{h}^{K-1}, \mathbf{h}^K; \rho_{dbn})$ is the joint distribution at the top-level RBM. Specifically, as illustrated in Fig.5 (a), learning the paraments $\rho = (\rho_{dbn}, \rho_{reg})$ of each detector includes two stages: first, $\rho_{dbn} = \{\mathbf{W}^i, \mathbf{u}^i, \mathbf{z}^i\}_{i=1..3}$ are estimated by pre-training the DBN using three RBMs layer-wisely. Then, we randomly initialize $\rho_{reg} = (\mathbf{W}^r, \mathbf{u}^r)$, and the initialized parameters ρ_{reg} along with the pre-trained parameters ρ_{dbn} are fine-tuned by logistic regression. This logistic regression layer maps the outputs of the last hidden layer to class labels and is optimized by minimizing the loss function between label hypothesis ($\widetilde{\ell}$) and ground truth ($\overline{\ell}$). In the following, we discuss how to estimate ρ_{dbn} by RBM in details.

As the building block of DBN (see Fig.5 (a)), a RBM is an undirected two-layer graphical model with hidden units ($\mathbf{h}$) and input units ($\mathbf{I}_b$). There are symmetric connections (*i.e.*, weights $\mathbf{W}$) between the hidden and visible units, but

[8] No need to evaluate the scores related to them. This is different from [8], where all part filters are visible.

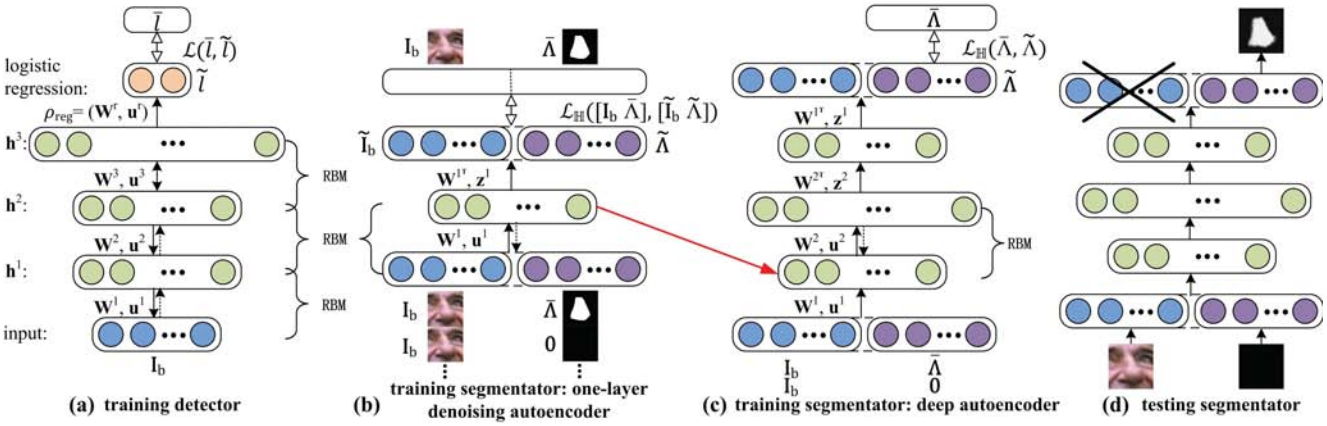

Figure 5. Illustration of learning process. We employ a four-layer DBN to model our detectors (a). The DBN is trained by layer-wise RBMs and tuned by logistic regression. Then, we propose a deep training strategy containing two steps to train the segmentators: first, we train a deep autoencoder (c), whose first layer are replaced by a one-layer denoising autoencoder (b). The deep autoencoder is tuned in one modality, while the one-layer autoencoder is tuned in both modalities. Each trained segmentator can directly output the label map using an image patch as input (d).

no connections within them. It defines a marginal probability over $\mathbf{I}_b$ using an energy model as below,

$$p(\mathbf{I}_b;\rho_{dbn}) = \sum_{\mathbf{h}} \frac{\exp\{\mathbf{z}^T\mathbf{I}_b + \mathbf{u}^T\mathbf{h} + \mathbf{h}^T\mathbf{W}\mathbf{I}_b\}}{Z}, \quad (7)$$

where Z is the partition function and $\mathbf{z}$, $\mathbf{u}$ are the offset vector for input units and hidden units respectively. In our case, the conditional probabilities of $p(\mathbf{h}|\mathbf{I})$ and $p(\mathbf{I}|\mathbf{h})$ can be simply modeled by products of Bernoulli distributions:

$$\begin{aligned} p(\mathbf{h}_i = 1|\mathbf{I}) &= sigm(\mathbf{u}_i + \mathbf{W}_{i\cdot}\mathbf{I}), \\ p(\mathbf{I}_j = 1|\mathbf{h}) &= sigm(\mathbf{z}_j + \mathbf{W}_{\cdot j}^T\mathbf{h}). \end{aligned} \quad (8)$$

$sigm(\cdot)$ is the *sigmoid* function. The parameters ρ_{dbn} can be estimated by taking gradient steps determined by the contrastive divergence [10].

3.3. Learning Segmentators

In this section, we introduce a deep learning approach for training the component segmentators, which transform image patches to label maps. The data transformation problem has been well examined by deep architectures (*i.e.*, multilayer network) in previous methods. Nevertheless, they mainly focus on single modality, such as deep autoencoder [11] and deep denoising autoencoder [24]. The former one encodes high-dimensional data into low-dimensional code and decodes the original data from it, while the latter one learns a more robust encoder and decoder which can recover the data even though they are heavily corrupted by noise. By combining and extending the existing works, we propose a deep training strategy containing two portions: we train 1) a deep autoencoder, whose first layer is replaced by 2) a one-layer denoising autoencoder. In the following, we explain how to generate the training data first, and then discuss the above two steps in detail.

To learn a mapping from images to label maps, we must explore the correlations between them. Therefore, unlike sec.3.2 where only image data are used for training, here we concatenate the images and ground truth label maps together as a training set. Since our purpose is to output label map given only image as input, we augment this training set by adding samples that have zero values of the label map and original values of the image (see Fig.5 (b)(c)). In other words, half of the training data has only image (*i.e.*, $(\mathbf{I}_b, \mathbf{0})$), while the other half has both image and ground truth label map (*i.e.*, $(\mathbf{I}_b, \overline{\Lambda})$). Similar strategy is adopted by [18], which learns features by using data from different modalities in order to improve the performance of classification performed in single modality.

1) Deep Autoencoder. As shown in Fig.5 (c), we establish a four-layer deep autoencoder, whose parameters ρ_{ae} can be defined as $\{\mathbf{W}^i, \mathbf{u}^i, \mathbf{z}^i\}_{i=1..2}$. The weights and offset vector for the first layer are achieved by a one-layer denoising autoencoder introduced in step 2). We estimate the weights of the second layer by RBM, and the weights of the upper two layers are tied similar to [11] (*i.e.*, $\mathbf{W}^3 = \mathbf{W}^{2^T}$, $\mathbf{W}^4 = \mathbf{W}^{1^T}$). Then, the whole network is tuned in single modality, that is minimizing the cross-entropy error $\mathcal{L}_{\mathbb{H}}$ between the outputs at the top layer (*i.e.*, reconstructed label maps $\widetilde{\Lambda}$) and the targets (*i.e.*, ground truth label maps $\overline{\Lambda}$).

2) One-layer denoising autoencoder. Modeling the low-level relations between data from different modalities is crucial but not a trivial task. Therefore, to improve the performance of the deep network, we specially learn its first layer with a denoising autoencoder [24] as shown in Fig.5 (b). Such shallow network is again pre-trained by RBM, but tuned in both modalities (*i.e.*, images and label maps). Note that in the fine-tuning stage, only the images and the ground truth label maps are used as the targets as shown at the top layer of Fig.5 (b).

Overall, with these two steps, each component segmentator learns a highly non-linear mapping from images to label maps. The testing procedure is illustrated in Fig.5 (d), where we delete the unused image data in the output. Thus, the deep network indeed outputs a label map given an image patch.

3.4. Implementation Details

In this section, we sketch several details for the sake of reproduction.

Training Detectors. We randomly select $3,500$ images from the Labeled Faces in the Wild (LFW) [12] database. We then randomly perturb each extracted image patch (see red boxes in Fig.2 (a)(b)) by translation, rotation, and scaling. Therefore, for each category, we totaly have $42,000$ training patches. Multi-label classification strategy is employed so that training three DBNs are enough (*i.e.*, one for face detector, one for part detectors, and the other for component detectors). More specifically, as shown in Fig.5 (a), we construct a 3-layers deep architecture and the unit numbers are 2 times, 3 times, and 4 times of the input size respectively. We supervisedly tune three DBNs with 2 outputs, 5 outputs, and 7 outputs at the top layer respectively. To construct the examples of the background category, we crop $105,000$ patches from the background for each DBN. All examples of the three DBNs are normalized to 64×64, 64×32, and 32×32 respectively. We use 9 gradient orientations and 6×6 cell size to extract the HOG feature.

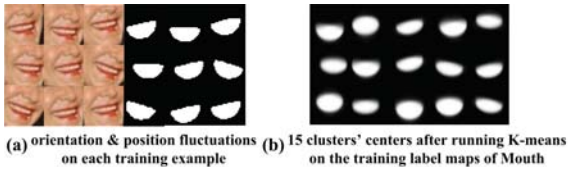

Figure 6. (a) Different translations and orientations are imposed in our training data. (b) shows how to deal with pose variations. We first separate the training data by K-means, then learn one deep autoencoder on each cluster.

Training Segmentators. We choose anther 500 images from the LFW [12] database, whose label maps are manually annotated. Nevertheless, in order to cover variant poses, we import fluctuations on position and orientation for each example as illustrated in Fig.6 (a). Similarly, all training patches are fixed at 32×32 and described by HOG feature. In order to account for pose variations, we train a set of 4-layers deep autoencoders for each component, which is obtained by first applying K-means on the label maps and then training one autoencoder on each cluster. Empirically, we set $K = 15$ (Fig.6 (b)).

4. Experiments

We conduct four experiments to test our algorithm. First, we perform face parsing on the LFW [12] database to evaluate the performance of our segmentators; Second, we collect a dataset from internet and compare the face alignment results with a state-of-the-art method, the Component-based Discriminative Search (CBDS) [16]; Third, we compare with two leading approaches (*i.e.*, BoRMaN [23] and Extended ASM (eASM) [17]) of feature point extractions. This experiment is conducted on the BioID [13] database; Finally, to further evaluate the generalization power, we carry out a segmentation task on the CUHK Face Sketch FERET Database (CUFSF) [29]. Note that for all these experiments, our model is trained on the LFW as outlined in sec.3.4.

Experiment I: performance of segmentators. We test our segmentators with a 7-classes facial image parsing experiment, which is to assign each pixel a class label (*e.g.*, left/right eye, left/right brow, nose, mouth, and background). First, we randomly select 300 images from the LFW database, and all the label maps of these images are well-labeled by hand. Then, our data-driven greedy search algorithm is performed on these images to achieve the reconstructed label maps, from which we obtain the final results after hysteresis thresholding. Fig.7 (a) shows the confusion matrix, in which accuracy values are computed as the percentage of image pixels assigned with the correct class label. The overall pixel-wise labeling accuracy is 90.86%. It demonstrates the ability of the learned segmentators on the segmentation of facial components. Several parsing results are visualized in Fig.10 (a).

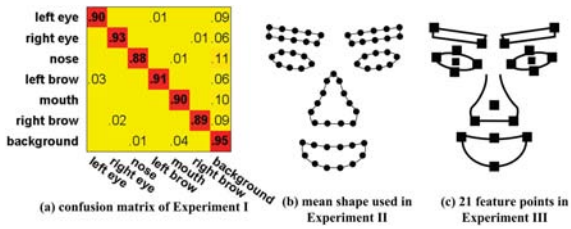

Figure 7. (a) shows the confusion matrix of Experiment I. (b) and (c) are the face models for Experiment II and Experiment III respectively.

Experiment II: face alignment. The purpose of our second experiment is to test our algorithm on the task of face alignment compared with the CBDS method, which was trained on many public databases including LFW over $4,000$ images. In the spirit of performing a database independent test, we collect a testing dataset containing 118 facial images from Google Image Search and Flickr. This dataset is challenging due to occlusion, background clutters, pose and appearance variations. Some examples are given in Fig.8.

To apply our method to face alignment, we first obtain the label map with the data-driven greedy search algorithm. Then a mean shape as illustrated in Fig.7 (b) is fitted to the label map by minimizing the Procrustes distance [7]. Note, the mean shape we used is different from the CBDS's. To allow a fair comparison, we thus exclude the bridge of the

Figure 8. Some samples in the testing dataset of Experiment II.

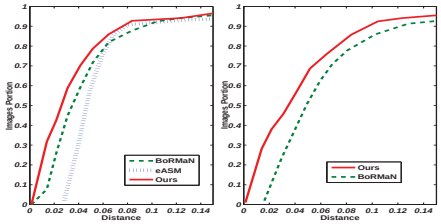

Figure 9. Comparisons of the cumulative error distribution of point-wise error measured on set A (left) and set B (right) respectively.

nose in our shape, and the inner lips and profiles of CBDS are also excluded. To yield a better alignment, we permit sightly non-rigid deformation of each landmark during the matching procedure. We summarize the results in Table 1, which shows the percentage of images with the root mean squared error (RMSE) less than the specific thresholds. As we can see, our algorithm achieves better results than CBDS. This is because our part-based detectors are very robust to occlusions and pose variations, and the segmentators can directly estimate the label map without a separated template matching process. The results of our method on several partially occluded faces are shown in Fig.10 (b).

Methods \ RMSE	<7 pixels	<9.5 pixels	<12 pixels
Ours	85.8%	92.5%	99.3%
CBDS [16]	76.9%	88.3%	98.1%

Table 1. Comparisons the alignment results with CBDS [16]. Each column is the percentage of images with RMSE less than a threshold.

Experiment III: feature point extraction. The goal of our third experiment is to compare our approach to BoRMaN and eASM with a facial feature points detection task. Fig.7 (c) plots the model of 21 feature points we use, which is defined similar to the BoRMaN method except the point at the facial profile. Our algorithm can be easily extended to consider the profile. Our feature points are extracted on the reconstructed label map, where two eye center points are equivalent to the centers of the eye detectors and the other boundary points are achieved by running fast corner detection. In this experiment, we collect two testing sets and denote them as A and B. Set A consists of all the original $1,521$ images in the BioID database, while set B consists of 200 randomly occluded images selected from set A. We separate this experiment into two parts: 1) we first compare with both BoRMaN and eASM on the set A, and 2) we compare with BoRMaN on the set B[9].

Part 1. Both BoRMaN and eASM evaluated their methods on the whole BioID database. We therefore can compare the performance with these two methods on the set A. The cumulated error distributions of the m_{e17} error measure [23] are illustrated in Fig.9 (left). The m_{e17} measure computes the mean error over all internal points, which are all the points except for the facial profile and eyelids. Fig.9 (left) shows that our approach outperforms these two methods.

Part 2. We then compare our method with BoRMaN on set B, where the images are partially occluded. The results are shown in Fig.9 (right). When occlusions are presented, we demonstrate that our approach provides significantly better results. Some results are plotted in Fig.10 (c).

Experiment IV: facial component segmentation. To further show that our algorithm can generalize to different facial modality, we conduct a 7-classes segmentation test on 100 face sketch images selected from the CUFSF database. The definition of the 7 classes is similar to Experiment I. We evaluate by computing the accuracy values, which are the percentage of image pixels assigned to the correct label. Several results are visualized in Fig.10 (d). Our overall pixel-wise labeling accuracy is 92.9%, which is better than 84.1% of the CBDS method.

5. Conclusion

In this paper, we propose a hierarchial face parser, where face parsing is achieved by part-based face detectors, component-based detectors, and component segmentators. For accurate face parsing, we recast segmentation of face components as the cross-modality data transformation problem, and solve it by a new deep learning strategy, which can output the label map given an image patch as input. By incorporating the deformable part-based detectors and the segmentators, our parser is very robust to occlusions, pose variations, and background clutters. We test our method on several applications and demonstrate great improvement.

6. Acknowledgement

This work is partially supported by Hong Kong SAR through RGC project 416510, and by Guangdong Province through Introduced Innovative R&D Team of Guangdong Province 201001D0104648280.

References

[1] B. Amberg, A. Blake, A. Fitzgibbon, S. Romdhani, and T. Vetter. Reconstructing high quality face-surfaces using model based stereo. *ICCV*, 2007.

[9] Since the program of eASM is not publicly obtainable, we only compare with BoRMaN on set B. Its executable program is available at: http://ibug.doc.ic.ac.uk/resources/facial-point-detector-2010/.

Figure 10. Demonstration of our results. The parsing results of Experiment I are shown in (a), where the 1^{st} column is the original images, and the $2^{nd}, 3^{rd}$ columns are the transformed label maps and the ground truth respectively. (b) is the aligned faces of Experiment II. (c) visualizes some results of facial feature extraction on the partially occluded set B. (d) is the segmentation results on face sketches.

[2] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. *ECCV*, 1998.

[3] T. Cootes, C. Taylor, and J. Graham. Active shape models their training and application. *Computer Vision and Image Understanding*, 1995.

[4] N. Cristianini and J. Shawe-Taylor. An introduction to support vector machines: and other kernel-based learning methods. *Cambridge University Press*, 2000.

[5] D. Cristinacce and T. Cootes. Boosted regression active shape models. *BMVC*, 2007.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *CVPR*, 2005.

[7] I. Dryden and K. Mardia. Statistical shape analysis. *John Wiley and Son, Chichester*, 1998.

[8] P. Felzenszwalb, R. Girshick, and D. McAllester. Cascade object detection with deformable part models. *CVPR*, 2010.

[9] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 2005.

[10] G. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 2006.

[11] G. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 2006.

[12] G. Huang, M. Ramesh, T. Berg, and E. Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. *Technical Report*, 2007.

[13] O. Jesorsky, K. Kirchberg, and R. Frischholz. Robust face detection using the hausdorff distance. *Lecture Notes in Computer Science*, 2001.

[14] L. Liang, F. Wen, X. Tang, and Y. Xu. An integrated model for accurate shape alignment. *ECCV*, 2006.

[15] L. Liang, F. Wen, Y. Xu, X. Tang, and H. Shum. Accurate face alignment using shape constrained markov network. *CVPR*, 2006.

[16] L. Liang, R. Xiao, F. Wen, and J. Sun. Face alignment via component-based discriminative search. *ECCV*, 2008.

[17] S. Milborrow and F. Nicolls. Locating facial features with an extended active shape model. *ECCV*, 2008.

[18] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng. Multimodal deep learning. *ICML*, 2011.

[19] X. Tang and X. Wang. Face sketch synthesis and recognition. *ICCV*, 2003.

[20] X. Tang and X. Wang. Face sketch recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 2004.

[21] J. Tu, Z. Zhang, Z. Zeng, and T. Huang. Face localization via hierarchical condensation with fisher boosting feature selection. *CVPR*, 2004.

[22] Z. Tu, X. Chen, A. Yuille, and S. Zhu. Image parsing: unifying segmentation, detection and recognition. *IJCV*, 2005.

[23] M. Valstar, B. Martinez, X. Binefa, and M. Pantic. Facial point detection using boosted regression and graph models. *CVPR*, 2010.

[24] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 2010.

[25] P. Viola and M. Jones. Robust real-time face detection. *IJCV*, 2004.

[26] X. Wang and X. Tang. Face photo-sketch synthesis and recognition. *TPAMI*, 2009.

[27] J. Warrell and S. Prince. Labelfaces: Parsing facial features by multiclass labeling with an epitome prior. *ICIP*, 2009.

[28] H. Wu, X. Liu, and G. Doretto. Face alignment via boosted ranking model. *CVPR*, 2008.

[29] W. Zhang, X. Wang, and X. Tang. Coupled information-theoretic encoding for face photo-sketch recognition. *CVPR*, 2011.

[30] Y. Zhou, W. Zhang, X. Tang, and H. Shum. A bayesian mixture model for multi-view face alignment. *CVPR*, 2005.