Deep Automatic Portrait Matting

Xiaoyong Shen[✉], Xin Tao, Hongyun Gao, Chao Zhou, and Jiaya Jia

The Chinese University of Hong Kong, Sha Tin, Hong Kong
xyshen@cse.cuhk.edu.hk
http://www.cse.cuhk.edu.hk/leojia/projects/automatting

Abstract. We propose an automatic image matting method for portrait images. This method does not need user interaction, which was however essential in most previous approaches. In order to accomplish this goal, a new end-to-end convolutional neural network (CNN) based framework is proposed taking the input of a portrait image. It outputs the matte result. Our method considers not only image semantic prediction but also pixel-level image matte optimization. A new portrait image dataset is constructed with our labeled matting ground truth. Our automatic method achieves comparable results with state-of-the-art methods that require specified foreground and background regions or pixels. Many applications are enabled given the automatic nature of our system.

Keywords: Portrait · Matting · Automatic method · Neural network

1 Introduction

Prevalence of smart phones makes self-portrait photography, i.e., selfie, possible whenever wanted. Accordingly image enhancement software gets popular for portrait beatification, image stylization, etc. to meet various aesthetic requirements. Interaction is a key component in many of these algorithms to draw strokes and select necessary areas. One important technique that is generally not automatic is image matting, which is widely employed in image composition and object extraction. Interaction is involved in existing systems to select foreground and background color samples using either strokes or regions.

Image matting takes a color image I as input and decomposes it into background B and foreground F assuming that I is blended linearly by F and B. Such composite can be expressed as

$$I = (1 - \alpha)B + \alpha F, \tag{1}$$

where α is the alpha matte for each pixel with range in $[0, 1]$. Since F, B and α are unknown, seven variables are to be estimated for each pixel, which makes the original matting problem ill-posed. Image matting techniques [1,2] require users specify foreground and background color samples with strokes or trimaps as shown in Fig. 1.

(a) Input   (b) Strokes   (c) Trimap   (d) AutoTrimap [3]

(e) Ours   (f) Matte of (b)   (g) Matte of (c)   (h) Matte of (d)

Fig. 1. Existing image matting methods need to specify background and foreground color samples. (b) and (c) show carefully created strokes and trimap. (f) and (g) show the corresponding closed-form matting results [1]. (d) is the trimap generated by automatic segmentation [3] followed by eroding the boundary for 50 pixels and (h) shows the corresponding closed-form matting result. (e) is our automatic matting result.

Problems of Interaction. Such interaction could be difficult for nonprofessional users without image matting knowledge. A more serious problem is that even with user-drawn strokes or regions, it is still not easy to know whether the color samples are enough or not before system optimization. As shown in Fig. 1(b) and (c), the human created strokes and trimap are already complicated, but the matting results by the powerful method [1] shown in (f) and (g) indicate the collected color samples are still insufficient. We note this is a very common problem even for professionals.

Statistically, 83.4 % chance is yielded to edit an image again after seeing the matting results in the first pass. On average 3.4 passes are needed to produce reasonable results on natural portrait images. Also the maximum number of passes to carefully edit a portrait image for color sample collection by us is 29. It shows a lot of effort has to be put to produce a reasonable alpha matte.

Importance and Difficulty of Automatic Matting. The above statistics lift the veil – automatic portrait image matting is essential for large-scale editing systems. It can relieve the burden for users to understand properties of color samples and the necessity to evaluate if they are enough for every local region.

Albeit fundamental, making an automatic matting system is difficult. The solution to compute the trimap intuitively from body segmentation may not generate good trimap with simple boundary erosion. One example is shown in Fig. 1(d), where the trimap by the automatic portrait segmentation method [3] results in the matte shown in (h).

Our Contribution. We propose a convolutional neural networks (CNNs) based system incorporating newly defined matting components. Although CNNs have demonstrated impressive success for a number of computer vision tasks such as detection [4–6], classification [7,8], recognition [9], and segmentation [10,11]. We cannot directly use existing structures for solving the matting problem since they learn hierarchical semantic features. For example, FCN [10] and CRFasRNN [11] have the ability to roughly separate the background and foreground. But they do not solve this problem without handling matting details.

For other low-level computer vision methods using CNNs for image super-resolution [12], deblurring [13] and filtering [14], mainly the powerful regression ability is made use of. They also do not fit our matting problem because no semantic information, such as human face and background scene, is considered.

Our network structure is novel on integrating two powerful functions. First, pixels are classified into background, foreground and unknown labels based on fully convolutional networks with several new components. For the second part, we propose the novel matting layer with forward and backward image matting formation. These two functions are incorporated in the unified end-to-end system without user interaction. Our method achieves decent performance for portrait image matting and benefits many tasks.

Further, we create a dateset including 2,000 portrait images, each with a full matte that involves all necessary details for training and testing.

2   Previous Work

We review natural image matting, as well as CNNs for pixel prediction related to our method.

2.1   Natural Image Matting

Natural image matting is originally ill-posed. To make the problem tractable, user specified strokes or trimap are used to sample foreground and background colors. There are quite a few matting methods, categorized according to color sampling and propagation. A survey is given in [15]. Quantitative benchmark is provided by Rhemann et al. [16].

Color Sampling Methods. Alpha values for two pixels can be close if the corresponding colors are similar. This rule motivates color sampling methods. Chuang et al. [17] proposed Bayesian matting by modeling background and

foreground color samples as Gaussian mixtures. Alpha values are solved for by using alternative optimization. Following methods include global color strategy [18], sample optimization [19], global sampling method [20], etc. Most color sampling methods need a high quality trimap, which is not easy to draw or refine.

Propagation Approaches. Another line is to propagate user-drawn information to unknown pixels according to pixel affinities. Levin et al. [1] developed closed-form matting by defining matting Laplacian under the color-line model. It updates to cluster-based spectral matting in [21]. To accelerate matting Laplacian computation, He et al. [22] computed the large-kernel Laplacian. Assuming intensity change locally smooth, Sun et al. [23] proposed Poisson image matting. The Laplacian affinities matrix can be constructed using nonlocal pixels. Following this principle, Chen et al. [2] developed the KNN matting. Since only sparse strokes are input to these systems, specifying them needs algorithm-level knowledge and the methods involve iterative update.

2.2   CNNs for Pixel Prediction

Semantic segmentation [24] has demonstrated the capability for predicting image pixel information. CNNs for segmentation are applied mainly in two ways. One is to learn image features and apply classification schemes to infer labels [25–27]. The other line is end-to-end learning from the image to the label map. Long et al. [10] designed a fully convolutional networks (FCN) for this task.

Directly regressing labels may lose edge accuracy. Recent work combines input image information to guide segmentation refinement, such as DeepLab [28], CRFasRNN [11], and deep parsing network [24]. Dai et al. [29] proposed box suppression. These CNNs for pixel prediction generate piece-wise constant label maps, which cannot be used for natural image matting.

3   Problem Understanding

Difficulties of automatic portrait image matting can be summarized in the following, facilitated by the illustrations in Fig. 2.

– **Rich Matte Details.** Portrait matting needs alpha values for all pixels and the matte is with rich details as shown in (d). These details often include hair with only several-pixel width, leading to difficult value prediction.
– **Ambiguous Semantic Prediction.** Portrait images have semantically meaningful structures in the foreground layer, such as eyes, hair, clothes, and mouth as shown in (e). Features are important to describe them.
– **Discrepant Matte Value.** There are only 5 % fractional values in the alpha matte, as shown in (c), with nearly 50 % of foreground semantic pixels that also create edges and boundaries. Such discrepancy often leads to inherent difficulty to estimate the small number of fractional alpha values.
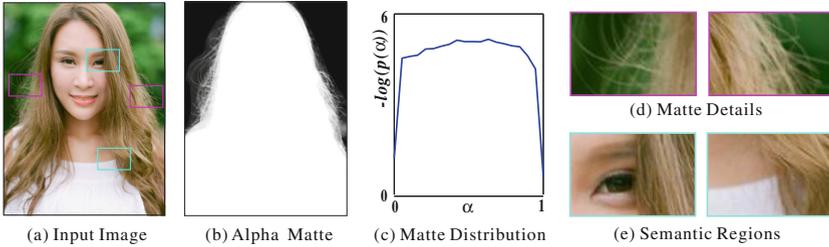
(a) Input Image   (b) Alpha Matte   (c) Matte Distribution   (d) Matte Details   (e) Semantic Regions

Fig. 2. An example to illustrate challenges. (a) and (b) are the input image and labeled alpha matte respectively. (c) is the alpha value distribution of (b) after negative-log transform. (d) are patches with matte details and (e) are semantic patches.

These issues make learning the alpha matte nontrivial. The CNNs for detection, classification and recognition are not concerned with image details. Segmentation CNNs [10] are with limited labels. Low-level task networks generally perform regression where the input and output are in the same domain of intensity or gradient. Crossing-domain inference from intensity to alpha matte is however considered in this paper.

4   Our Approach

We show the pipeline of our system in Fig. 3. The input is a portrait image I and the output is the alpha matte $\mathcal{A}$. Our network includes the trimap labeling and image matting modules.

4.1   Trimap Labeling

Each trimap includes foreground, background and unknown pixels. Our trimap labeling aims to predict the probability that each pixel belongs to these classes. As shown in Fig. 3, this part takes the input image and generates three channels F^s, B^s and U^s. Each value of a pixel stores the score for one channel. A large score indicates high probability in the corresponding class.

We model it as a pixel classification problem. We follow the FCN-8s setting [10] and incorporate special components for matting. The output is on the 3 aforementioned channels and one extra channel of *shape mask* for further performance improvement.

Shape Mask. The shape mask channel is shown in Fig. 3(a). It is based on the fact that a typical portrait includes head and part of shoulder, arm, and upper body. We thus include a channel, in which a subject region is aligned with the actual portrait. This is particularly useful since we explicitly provide feature to the network for reasonable initialization of the alpha matte.

To generate this channel, we compute an aligned average mask from our training data. For each training portrait-matte pair $\{P^i, M^i\}$ where P^i is the
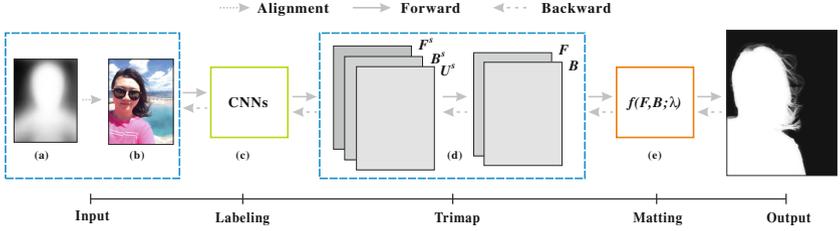
Fig. 3. Pipeline of our end-to-end portrait image matting network. It includes trimap labeling (c) and image matting (e). They are linked with forward and backward propagation functions.

feature point computing by face alignment [30] and M^i is the labeled alpha matte, we transform M^i using homography $\mathcal{T}_i$, estimated from the facial feature points of P^i and a face template. We compute the mean of these transformed mattes as

$$M = \frac{\sum_i m_i \cdot \mathcal{T}_i(M^i)}{\sum_i m_i}, \tag{2}$$

where m_i is a matrix with the same size as M^i, indicating whether the pixel in M^i is outside the image or not after transform $\mathcal{T}_i$. The value is 1 if the pixel is inside the image, otherwise it is 0. The operator $\cdot$ denotes element-wise multiplication. This shape mask M, which has been aligned to a portrait template, can then be similarly transformed for alignment with the facial feature points of the input portrait. The added shape mask helps reduce prediction errors. We will discuss its performance in our experiment section.

4.2   Image Matting Layer

With the output score channels F^s, B^s and U^s, we get the probability maps F and B for foreground and background respectively by a softmax function. The formulation for F is written as

$$F = \frac{\exp(F^s)}{\exp(F^s) + \exp(B^s) + \exp(U^s)}. \tag{3}$$

Similarly, we obtain the probability map B for background pixels. For convenience, F and B are expressed as vectors. Then the alpha matte can be computed through propagation as

$$\min \quad \lambda \mathcal{A}^T \mathbf{B} \mathcal{A} + \lambda (\mathcal{A} - \mathbf{1})^T \mathbf{F} (\mathcal{A} - \mathbf{1}) + \mathcal{A}^T \mathcal{L} \mathcal{A}, \tag{4}$$

where $\mathcal{A}$ is the alpha matte vector and $\mathbf{1}$ is an all-1 vector. $\mathbf{B} = \mathrm{diag}(B)$ and $\mathbf{F} = \mathrm{diag}(F)$. $\mathcal{L}$ is the matting Laplacian matrix [1] with respect to the input image I. λ is a parameter to balance the data term and the matting Laplacian.

According to the solution of Eq. (4), our image matting layer as shown in Fig. 3(e) can be expressed as

$$f(F, B; \lambda) = \lambda(\lambda\mathbf{B} + \lambda\mathbf{F} + \mathcal{L})^{-1}F, \tag{5}$$

where F and B are the input data and λ is the parameter for learning. $f(F, B; \lambda) = \mathcal{A}$ defines the forward process. As shown in Fig. 3, in order to combine the image matting layer with previous CNNs, one important issue is to back propagate errors. Each layer should provide the derivatives $\frac{\partial f}{\partial F}$, $\frac{\partial f}{\partial B}$ and $\frac{\partial f}{\partial \lambda}$ with respect to the input and parameters.

Claim. Partial derivatives of Eq. (5) with respect to B, F and λ are with the closed-form expression as

$$\frac{\partial f}{\partial B} = -\lambda^2 \mathbf{D}^{-1}\text{diag}(\mathbf{D}^{-1}F), \tag{6}$$

$$\frac{\partial f}{\partial F} = \frac{\partial f}{\partial B} + \lambda\mathbf{D}^{-1}, \tag{7}$$

$$\frac{\partial f}{\partial \lambda} = -\lambda\mathbf{D}^{-1}\text{diag}(F + B)\mathbf{D}^{-1}F + \mathbf{D}^{-1}F, \tag{8}$$

where $\mathbf{D} = \lambda\mathbf{B} + \lambda\mathbf{F} + \mathcal{L}$. $\frac{\partial f}{\partial B}$, $\frac{\partial f}{\partial F}$ and $\frac{\partial f}{\partial \lambda}$ vector-form derivatives. They can be efficiently computed by solving sparse linear systems.

Proof. Given $\partial(AB) = (\partial A)B + A(\partial B)$ and $\partial(A^{-1}) = -A^{-1}(\partial A)A^{-1}$, we get

$$\frac{\partial f}{\partial B} = -\lambda\mathbf{D}^{-1}\frac{\partial\mathbf{D}}{\partial B}\mathbf{D}^{-1}F + \lambda\mathbf{D}^{-1}\frac{\partial F}{\partial B}. \tag{9}$$

Now, since $\frac{\partial\mathbf{D}}{\partial B_i} = \text{diag}(\frac{\partial(\lambda B + \lambda F)}{\partial B_i})$, a matrix is formed with its ith diagonal element being λ and all others being zeros. Since the second term of Eq. (9) gives a zero matrix, this directly yields Eq. (6). With similar derivation, we produce Eqs. (7) and (8). Because $\mathbf{D}$ is a sparse 25 diagonal matrix [1] and $\mathbf{D}^{-1}F$ can be computed by solving the linear system $\mathbf{D}X = F$, all derivatives can be updated by solving sparse linear systems.

With these derivatives, the image matting layer can be added to the CNNs as shown in Fig. 3 for optimization using the forward and backward propagation strategy. The parameter λ, which balances the data term and matting Laplacian, is also adjusted during the training process. Note that it is manually tuned in previous work.

4.3   Loss Function

The loss function measures the error between the predicted alpha matte and ground truth. Generally, the errors are calculated as the L_2- or L_1-norm distance. But in our task, most pixels have 0 or 1 alpha values because solid foreground and background pixels are the majority as shown in Fig. 2.

Therefore, directly applying L_2- or L_1-norm measure will be biased to absolute background and foreground pixels, which is not what we want. We find that setting different weights to alpha values make the system more reliable. It leads to our final loss function as

$$L(\mathcal{A}, \mathcal{A}^{gt}) = \sum_i w(\mathcal{A}_i^{gt}) \|\mathcal{A}_i - \mathcal{A}_i^{gt}\|, \tag{10}$$

where $\mathcal{A}$ is the alpha matte to be measured and $\mathcal{A}^{gt}$ is the corresponding ground truth. i indexes pixel position. $w(\mathcal{A}_i^{gt})$ is the weight function, which we define according to the value distribution of ground truth mattes, written as

$$w(\mathcal{A}_i^{gt}) = -\log(p(A = \mathcal{A}_i^{gt})), \tag{11}$$

where A is the random variable for the alpha matte and $p(A)$ models its probability distribution. We compute $p(A)$ from our ground truth mattes, which will be detailed later. Note that such a loss function is essential for our framework because there are only 5 % pixels in the image with alpha values not 0 or 1.

4.4   Analysis

Our end-to-end network for portrait image matting directly learns the alpha matte from the input image. We incorporate the trimap channel as a layer before image matting, as shown in Fig. 3. This setting is better than straightforward learning the trimap. We analyze it from our back-propagation process. We denote the total loss from F and B to the ground truth $\mathcal{A}^{gt}$ as $L^*(F, B, \lambda; \mathcal{A}^{gt})$. With the back-propagation formula, its derivative according to B is expressed as

$$\frac{\partial L^*(F, B, \lambda; \mathcal{A}^{gt})}{\partial B} = \frac{\partial L(\mathcal{A}, \mathcal{A}^{gt})}{\partial \mathcal{A}} \frac{\partial f(F, B; \lambda)}{\partial B}, \tag{12}$$

where $L(\mathcal{A}, \mathcal{A}^{gt})$ and $f(F, B; \lambda)$ are the loss function and matting function defined in Eqs. (10) and (5) respectively. $\mathcal{A}$ is the output value of $f(F, B; \lambda)$. Since
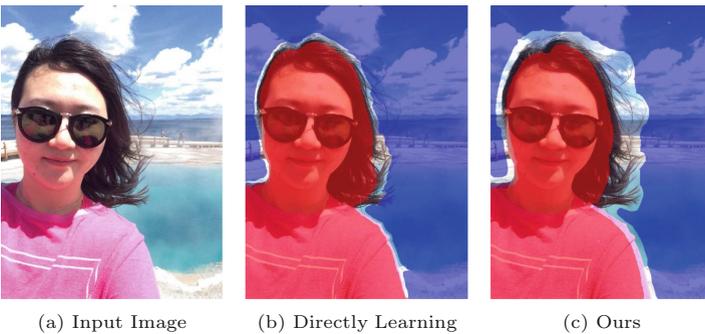

(a) Input Image   (b) Directly Learning   (c) Ours

Fig. 4. Trimap comparison between directly learning and learning in our end-to-end framework.

$\frac{\partial f(F,B;\lambda)}{\partial B}$ (defined in Eq. (6)) is related to the matting Laplacian $\mathcal{L}$, the loss $L^*(F, B, \lambda; \mathcal{A}^{gt})$ is related to not only the alpha matte loss $L(\mathcal{A}, \mathcal{A}^{gt})$ but also the matting function $f(F, B; \lambda)$. This indicates that the predicted trimap is optimized according to the matting scheme, and explains why such setting outperforms direct trimap learning.

To demonstrate it, we conduct experiments based on the model that only includes the trimap labeling part. In the training process, the ground truth trimap is obtained according to the alpha matte, where we set pixels with values between 0 and 1 as unknown ones. We compare the trimap results of this naive system with our complete ones. As shown in Fig. 4(b), directly learning the trimap makes hair predicted as background. Our complete system, as shown in (c), addresses this problem.

5   Data Preparation and Training

We provide new training data to appropriately learn the model for portrait image matting.

Dataset. We collected portrait images from Flickr. They are then selected to make sure portraits are with a good variety of age, color, clothing, accessories, hair style, head position, background scene, etc. The matting regions are mainly around hair and soft edges caused by depth-of-field. All images are cropped such that the face rectangles are with similar sizes. Several examples are shown in Fig. 5.

With the selected portrait images, we create alpha mattes with intensive user interaction to make sure they are with high quality. First, we label the

Fig. 5. Images in our dataset. They are with large structure variation for both foreground and background regions.

trimap of each image by zoom-in into local areas. Then we compute mattes using closed-form matting [1] and KNN matting [2]. The two computed mattes for each image overlay a background image for manually inspecting the quality. We choose the better one for our dataset. The result is discarded if both mattes cannot meet our high standard. When necessary, small errors are remedied by Photoshop [31]. After this labeling process, we collect 2,000 images with high-quality mattes. These images are randomly split into the training and testing sets with 1,700 and 300 images respectively.

Model Training. We augment the number of images by perturbing them with rotation and scaling. Four rotation angles $\{-45°, -22°, 22°, 45°\}$ and four scales $\{0.6, 0.8, 1.2, 1.5\}$ are used. We also apply four different Gamma transforms to increase color variation. The Gamma values are $\{0.5, 0.8, 1.2, 1.5\}$. After these transforms, we have 16K+ training images. The variation we introduce greatly improves the performance of our system to handle new images with possibly different scale, rotation and tone.

We set our model training and testing on the Caffe platform [32]. With the model illustrated in Fig. 3, we implement the image matting layer and loss layer as new components in the system. To efficiently solve the sparse linear system defined in the forward and back-propagation phases, we apply Intel MKL Parallel Direct Sparse Solver (PARDISO). The widely used SGD solver is adopted to optimize our model during training.

We initialize parameters using the FCN-8s model. Since our network only outputs three channels, we randomly select their parameters from the original 21 channels for initialization. For the matting layer, we set the initial λ to 100. We tune the learning rates in range $[1e-3, 1e-6]$. For each learning rate, we analyze the loss change and test the performance.

Running Time. We conduct training and testing on a single NVIDIA Titan X graphics card. Our model training phase requires about 20 epochs. The training time is about one day. For the testing phase, the running time on a 600×800 color image is 0.6 second. Our testing conducted on CPU takes about 6 seconds using the Intel MKL-optimized Caffe.

6 Experiments and Results

We show the performance of our system and perform comparison. Applications related to our automatic portrait image matting are also presented.

6.1 Performance Evaluation

We first evaluate our method using the testing dataset. Comparison with baselines is analyzed below.

Accuracy Measure. We follow the matte perceptual error [16] to measure matting quality. The two types of errors are *gradient* and *connectivity* ones expressed as

$$G(\mathcal{A}, \mathcal{A}^{gt}) = \frac{1}{K} \sum_i \|\nabla \mathcal{A}_i - \nabla \mathcal{A}_i^{gt}\|, \tag{13}$$

$$C(\mathcal{A}, \mathcal{A}^{gt}) = \frac{1}{K} \sum_i \|\varphi(\mathcal{A}_i, \Omega) - \varphi(\mathcal{A}_i^{gt}, \Omega\|, \tag{14}$$

where $\mathcal{A}$ is the matte and $\mathcal{A}^{gt}$ is the corresponding ground truth. K is the number of pixels in $\mathcal{A}$. ∇ is the operator to compute gradients. $\varphi(\mathcal{A}_i, \Omega)$ measures connectivity of the matte in pixel i regarding neighborhood Ω [16].

Methods Comparison. We compare several automatic schemes as baselines for matting. The first one is automatic trimap generation based on segmentation from graph cuts [33]. To collect seeds, the face tracker [30] is employed to produce face feature points; then color samples around these points are set as foreground seeds. The background seeds are randomly sampled. Finally, the trimap is computed by eroding segmentation result and the final alpha matte is optimized by closed-form matting. We test different eroding widths and parameters in matting and choose the best result for comparison. Similar to graph-cut segmentation, we also generate trimap from the automatic portrait segmentation method [3] and compute the closed-form matting result.

The other three baselines are achieved by learning the trimap via semantic segmentation FCN [10], DeepLab [28] and CRFasRNN [11]. To adapt these frameworks to the matting problem, we modify the output channel number to three for background, foreground and unknown pixels. The trimap ground truth is computed from our training dataset by setting the alpha matte values between 0 and 1 as unknown. After getting the trimap, parameter-tuned closed-form matting is conducted to compute the final alpha matte. Parameters for learning are also adjusted according to the matting performance.

We report the performance of these automatic matting methods in Table 1. Several examples are shown in Fig. 6. The results indicate that the graph-cut

Table 1. Accuracy of our method and four other automatic matting baselines on our testing dataset. Usefulness of the shape mask is also verified.

Methods	Grad. Error ($\times 10^{-3}$)	Conn. Error ($\times 10^{-4}$)
Graph-cut trimap	4.93	7.73
Trimap by [3]	4.61	7.63
Trimap by FCN [10]	4.14	7.61
Trimap by DeepLab [28]	3.91	7.52
Trimap by CRFasRNN [11]	3.56	7.39
Ours without shape mask	3.11	6.99
Ours	**3.03**	**6.90**

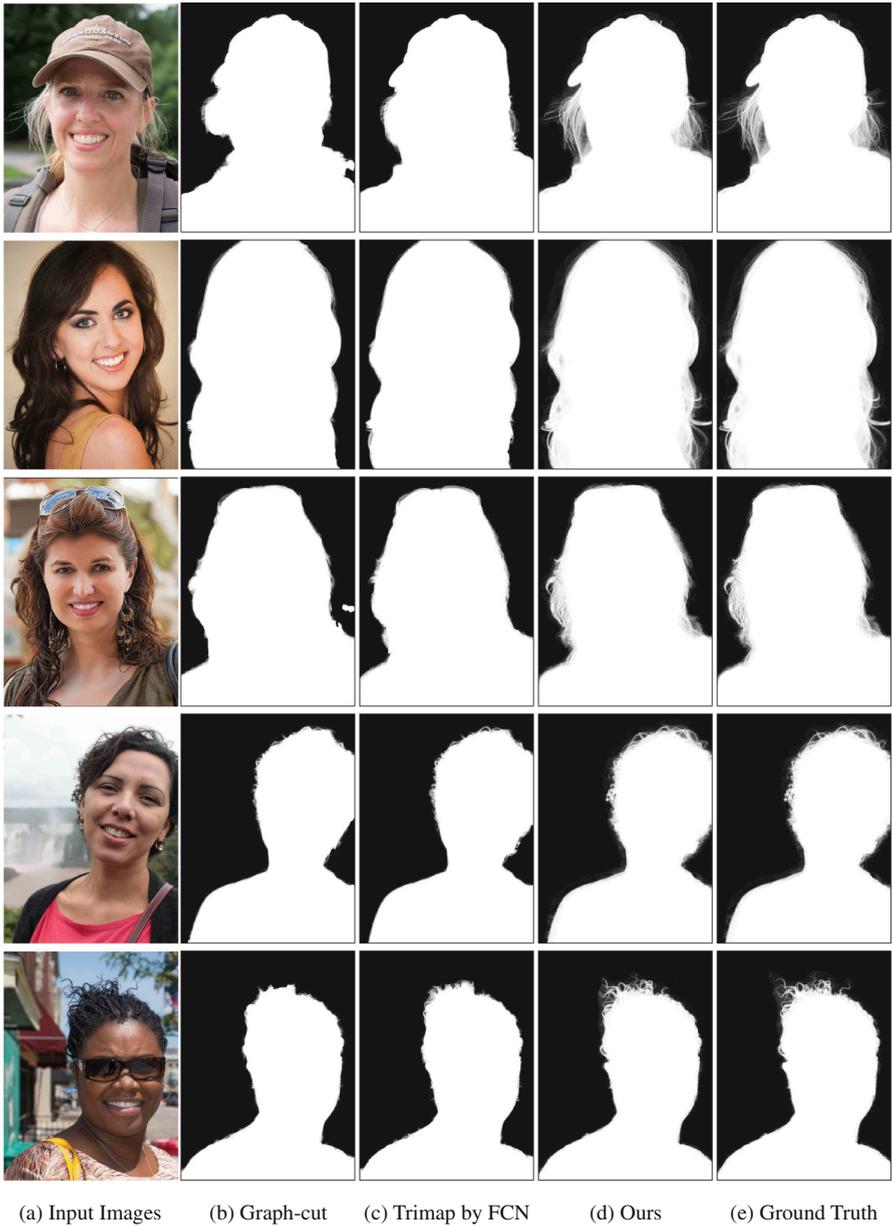(a) Input Images   (b) Graph-cut   (c) Trimap by FCN   (d) Ours   (e) Ground Truth

Fig. 6. Visual comparison. (a) Input images. (b) Graph-cut trimap baseline results. (c) Trimaps by FCN baseline. (d) Our results. (e) Ground truth mattes.

based baseline does not work well because the segmentation is with large errors. Although the automatic portrait segmentation method [3] provides nice segmentation results, simple boundary erosion cannot generate high-quality matting as reported in Table 1.

The methods based on trimap learning perform better. The errors, as shown in Fig. 6(c), are mainly caused by lose of matte details in the learned trimap. It is, as aforementioned, because the semantic frameworks are designed to produce less accurate segment information. Our results are shown in Fig. 6(d), which are with the best quality. Our end-to-end CNN not only has the ability to infer the background, foreground, and unknown regions, but also adjusts the trimap via the proposed matting layer.

Effectiveness of Shape Mask. The shape mask is taken as an additional input channel in our system. It can actually reduce matting errors for pixels far from the portrait region. Such errors are ubiquitous when similar structures exist in the background scene. As shown in Fig. 7, the result without the shape mask contains noticeable errors in the top right corner in (b) while the result by our complete system does not have this problem in (c). Quantitative evaluation is reported in Table 1.
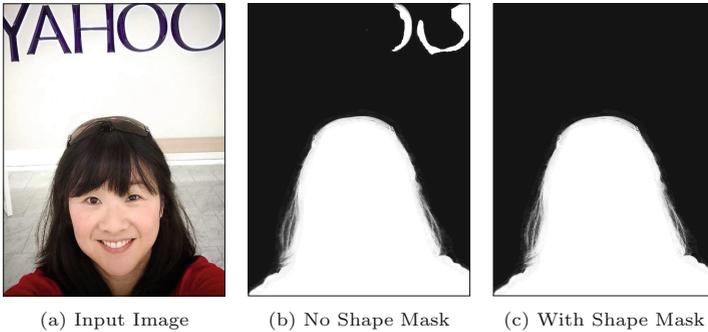

<div align="center">
(a) Input Image   (b) No Shape Mask   (c) With Shape Mask
</div>

Fig. 7. Trimap comparison between directly learning and learning in our end-to-end framework.

6.2   Applications

A number of applications are enabled due to the fully automatic mechanism of our matting system. In Fig. 8, the alpha matte in a portrait image is automatically generated by our method. They are ready for fast stylization, color transform, depth-of-field and background editing. Respective effects are shown from (c)–(h). More results are provided in our supplementary material.
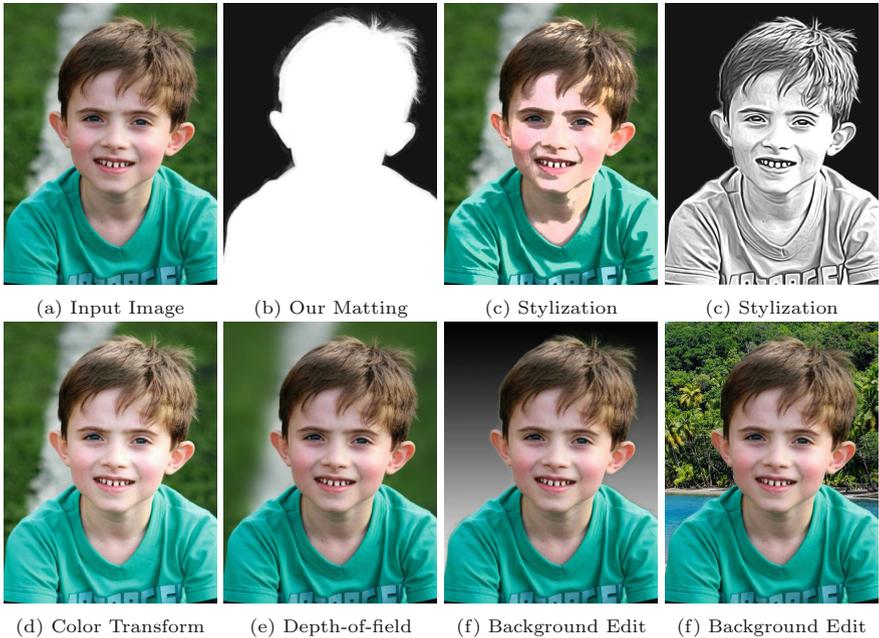
(a) Input Image   (b) Our Matting   (c) Stylization   (c) Stylization

(d) Color Transform   (e) Depth-of-field   (f) Background Edit   (f) Background Edit

Fig. 8. Applications of our automatic portrait matting. (a) and (b) are the input image and our automatic matting result. (c)–(h) show different editing effects.
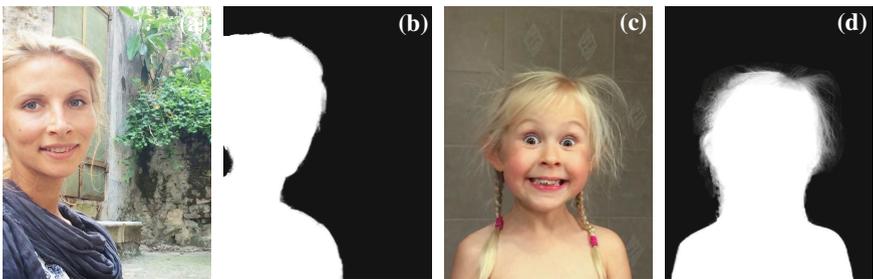

Fig. 9. Failure cases. The first example in (a) shows very low contrast between the foreground and background. The second image in (c) is with complicated hair structure. (b) and (d) are our results respectively.

7 Conclusion and Limitations

We have proposed a fully automatic matting system for portrait photos. It is based on end-to-end CNNs with several new components, including trimap labeling, shape mask incorporation, and matting layer design. A new portrait dataset containing images and their corresponding high quality alpha mattes is

constructed for system training and testing. Our method achieves decent performance without any user interaction.

There are inevitable limitations. First, when contrast between background and foreground is rather low or the unknown region is quite large, our method may not work well. One example is shown in Fig. 9(a), which the color of hair is very similar to background. Second, complicated structure on foreground hair or background scene that is not common in images could also adversely influence our method. One example is given in Fig. 9(c).

References

1. Levin, A., Lischinski, D., Weiss, Y.: A closed-form solution to natural image matting. IEEE Trans. Pattern Anal. Mach. Intell. **30**(2), 228–242 (2008)
2. Chen, Q., Li, D., Tang, C.: KNN matting. IEEE Trans. Pattern Anal. Mach. Intell. **35**(9), 2175–2188 (2013)
3. Shen, X., Hertzmann, A., Jia, J., Paris, S., Price, B., Shechtman, E., Sachs, I.: Automatic portrait segmentation for image stylization. In: Eurographcis (2016)
4. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat: Integrated recognition, localization and detection using convolutional networks. CoRR abs/1312.6229 (2013)
5. Girshick, R.B., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR, pp. 580–587(2014)
6. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans. Pattern Anal. Mach. Intell. **37**(9), 1904–1916 (2015)
7. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS, pp. 1106–1114 (2012)
8. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR, pp. 1–9 (2015)
9. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556 (2014)
10. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR (2014)
11. Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., Torr, P.H.S.: Conditional random fields as recurrent neural networks. In: ICCV (2015)
12. Dong, C., Loy, C.C., He, K., Tang, X.: Image super-resolution using deep convolutional networks. In: ECCV (2014)
13. Xu, L., Ren, J.S.J., Liu, C., Jia, J.: Deep convolutional neural network for image deconvolution. In: NIPS, pp. 1790–1798 (2014)
14. Xu, L., Ren, J., Yan, Q., Liao, R., Jia, J.: Deep edge-aware filters. In: ICML, pp. 1669–1678 (2015)
15. Wang, J., Cohen, M.F.: Image and video matting: a survey. Found. Trends Comput. Graph. Vis. **3**(2), 97–175 (2007)

16. Rhemann, C., Rother, C., Wang, J., Gelautz, M., Kohli, P., Rott, P.: A perceptually motivated online benchmark for image matting. In: CVPR, pp. 1826–1833 (2009)
17. Chuang, Y., Curless, B., Salesin, D., Szeliski, R.: A bayesian approach to digital matting. In: CVPR, pp. 264–271 (2001)
18. Wang, J., Cohen, M.F.: An iterative optimization approach for unified image segmentation and matting. In: CVPR, pp. 936–943 (2005)
19. Wang, J., Cohen, M.F.: Optimized color sampling for robust matting. In: CVPR (2007)
20. He, K., Rhemann, C., Rother, C., Tang, X., Sun, J.: A global sampling method for alpha matting. In: CVPR, pp. 2049–2056 (2011)
21. Levin, A., Rav-Acha, A., Lischinski, D.: Spectral matting. IEEE Trans. Pattern Anal. Mach. Intell. **30**(10), 1699–1712 (2008)
22. He, K., Sun, J., Tang, X.: Fast matting using large kernel matting laplacian matrices. In: CVPR, pp. 2165–2172 (2010)
23. Sun, J., Jia, J., Tang, C., Shum, H.: Poisson matting. ACM Trans. Graph. **23**(3), 315–321 (2004)
24. Liu, Z., Li, X., Luo, P., Loy, C.C., Tang, X.: Semantic image segmentation via deep parsing network. In: ICCV (2015)
25. Arbelaez, P., Hariharan, B., Gu, C., Gupta, S., Bourdev, L.D., Malik, J.: Semantic segmentation using regions and parts. In: CVPR, pp. 3378–3385 (2012)
26. Mostajabi, M., Yadollahpour, P., Shakhnarovich, G.: Feedforward semantic segmentation with zoom-out features. In: CVPR (2014)
27. Farabet, C., Couprie, C., Najman, L., LeCun, Y.: Learning hierarchical features for scene labeling. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1915–1929 (2013)
28. Chen, L., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Semantic image segmentation with deep convolutional nets and fully connected crfs. In: ICLR (2014)
29. Dai, J., He, K., Sun, J.: Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In: CVPR (2015)
30. Saragih, J.M., Lucey, S., Cohn, J.F.: Face alignment through subspace constrained mean-shifts. In: ICCV, pp. 1034–1041 (2009)
31. Systems, A.: Adobe photoshop cc 2015 tutorial
32. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093 (2014)
33. Boykov, Y.Y., Jolly, M.P.: Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In: ICCV, vol. 1, pp. 105–112 (2001)