Face Sketch Synthesis with Style Transfer using Pyramid Column Feature

Anonymous ICCV submission

Paper ID 612

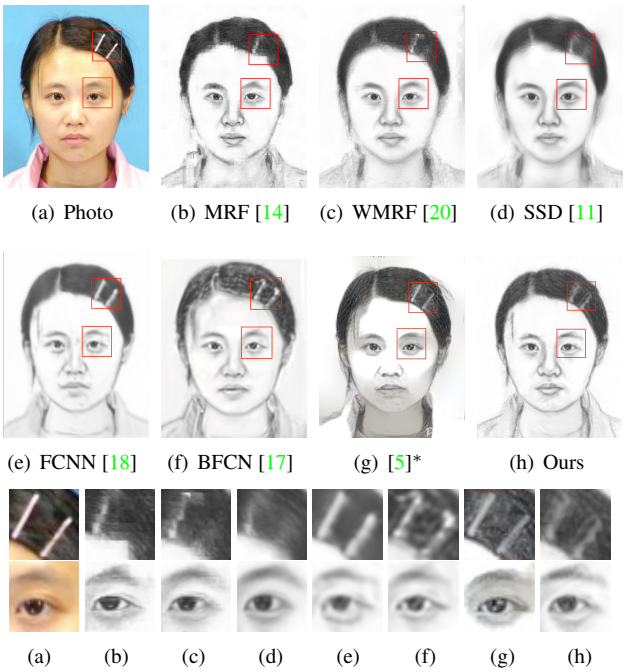
Abstract

In this paper, we propose a novel framework based on deep neural networks for face sketch synthesis from a photo. Imitating the process of how artists draw sketches, our framework synthesizes face sketches in a cascaded manner. A content image is first generated that outlines the shape of the face and the key facial features. Textures and shadings are then added to enrich the details of the sketch. We utilize a fully convolutional neural network (FCNN) to create the content image, and propose a style transfer approach to introduce textures and shadings based on a newly proposed pyramid column feature. We demonstrate that our style transfer approach based on the pyramid column feature can not only preserve more sketch details than the common style transfer method, but also surpasses traditional patch based methods. Quantitative and qualitative evaluations suggest that our framework outperforms other state-of-the-arts methods, and can also generalize well to different test images.

1. Introduction

Face sketch synthesis has drawn a great attention from the community in recent years because of its wide range of applications. For instance, it can be exploited in law enforcement for identifying suspects from a mug shot database consisting of both photos and sketches. Besides, face sketches have also been widely used for entertainment purpose. For example, filmmakers could employ face sketch synthesis technique to ease the cartoon production process.

Unfortunately, there exists no easy solution to face sketch synthesis due to the big stylistic gap between photos and sketches. In the past two decades, a number of exemplar based methods [14, 11, 19, 20] were proposed. In these methods, a test photo is first divided into patches. For each test patch, a candidate sketch patch is identified by finding the most similar photo patch in a training set of photo-sketch pairs. The main drawback of this approach is that if there exists no photo patch in the training set which is sufficiently similar to a test patch, loss of content will be observed in the synthesized sketch. For example, the sketches in the first row of Fig. 1 fail to keep the hairpins. Besides, some methods [11, 20] blur away the textures when they try to eliminate the inconsistency between neighboring patches. Another common problem is that the synthesized sketch may not look like the test photo (see the left eye in Fig. 1(b)). Recently, approaches [17, 18] based on convolutional neural network (CNN) were developed to solve these problems. Since they directly generate sketches from photos, structures and contents of the photos can be maintained. However, the pixel-wise loss functions adopted



(a) Photo (b) MRF [14] (c) WMRF [20] (d) SSD [11]

(e) FCNN [18] (f) BFCN [17] (g) [5]* (h) Ours

(a) (b) (c) (d) (e) (f) (g) (h)

Figure 1. Face sketches generated by existing methods and the proposed method. Our method can not only preserve both hair and facial content, but also contains sharp textures. (Note that (g) is obtained from the deep art website[1] using the photo as content and a sketch from the training set as style.)

[1] https://deepart.io/

by these methods will lead to blurry artifacts (see Fig. 1(e) and 1(f)) because they are incapable of preserving texture structures. The popular neural style transfer provides a better solution for texture synthesis. However, there exist two obstacles in directly applying such a technique. First, the result is easily influenced by the illumination of the photo (see the face in Fig. 1(g)). Second, it requires a style image to provide the global statistics of the textures. If the given style image does not match with the target sketch (which we do not have), some side effects will occur (see the nose in Fig. 1(g)). *(ken: I cannot see the problem in 1(h))*

For an artist, the process of sketching a face usually starts with outlining the shape of the face and the key facial features like the nose, eyes, mouth and hair. Textures and shadings are then added to regions such as hair, lips, and bridge of the nose to give the sketch a specific style. Based on the above observation, and inspired by neural style transfer [4], we propose a new framework for face sketch synthesis from a photo that overcomes the aforementioned limitations. In our method, a content image that outlines the face is generated by a feed-forward neural network, and textures and shadings are then added using a style transfer approach. Specifically, we design a new architecture of fully convolutional neural network (FCNN) composed of inception layers [12] and convolution layers with batch normalization [6] to generate the content image (see Section 4.1). To synthesize the textures, we first divide the target sketch into a grid. For each grid cell, we compute a newly proposed pyramid column feature using the training set (see Section 4.2). A target style can then be computed from a grid of these pyramid column features, and applied to the content image. Our approach is superior to the current state-of-the-art methods in that

- It is capable of generating more stylistic sketches without introducing over smoothing artifacts.
- It can preserve the content of the test photo well.

2. Related Work

2.1. Face Sketch Synthesis

Based on the taxonomy of previous studies [11, 20], face sketch synthesis methods can be roughly categorized into profile sketch synthesis methods [1, 2, 15] and shading sketch synthesis methods [8, 11, 13, 14, 18, 19, 20]. Compared with profile sketches, shading sketches are more expressive and thus more preferable in practice. Based on the assumption that there exists a linear transformation between a face photo and a face sketch, the method in [13] computes a global eigen-transformation for synthesizing a face sketch from a photo. This assumption, however, does not always hold since the modality of face photos and that of face sketches are quite different. Liu et al. [8] pointed out that the linear transformation holds better locally, and therefore they proposed a patch based method to perform sketch synthesis. In [14], a MRF based method was proposed to preserve large scale structures across sketch patches. Variants of the MRF based method were introduced in [19, 20] to improve the robustness to lighting and pose, and to render the ability of generating new sketch patches. In addition to these MRF based methods, approaches based on guided image filtering [11] and feed-forward convolutional neural network [18] are also found to be effective in transferring photos into sketches. A very recent work similar to ours is reported by Zhang *et al*. [17]. They proposed a two-branch FCNN to learn content and texture respectively, and then fused them through a face probability map. Although their results are impressive, their sketch textures do no look natural and the facial components are over smoothed.

2.2. Style Transfer with CNN

Texture synthesis has long been a challenging task. Traditional methods can only imitate repetitive patterns. Recently, Gatys *et al*. [4, 5] studied the use of CNN in style representation, and proposed a method for transferring the style of one image (referred to as the style image) to another (referred to as the content image). In their method, a target style is first computed based on features extracted from the style image using the VGG-Network. An output image is then generated by iteratively updating the content image and minimizing the difference between its style and the target style. Justin *et al*. [7] further accelerated this process by learning a feed forward CNN in the training stage. These methods represent styles by a multi-scale Gram matrix of the feature maps. Since the Gram matrix only cares about global statistics, local structures may be destroyed when the style image is very different from the content image. Although this may not be a problem in transferring artistic styles to images, this will definitely produce noticeable artifacts in the face sketch as people are very sensitive to the distortions of the facial features. In [3], Chen and Schmidt proposed a different patch based style transfer method which is better at capturing local structures. However, it is still far from satisfactory to be employed in face sketch synthesis. Our style transfer approach is inspired by but different from the above work [4, 5, 7] in that our target style is computed from image patches of many different images rather than from just one single image.

3. Style Representation

Following the work of [5], we use Gram matrices of VGG-19 [10] feature maps as our style representation. Denote the vectorized cth channel of the feature map in the lth layer of the final sketch $\mathcal{X}$ by $F_c^l(\mathcal{X})$. A Gram matrix of the feature map in the lth layer is then defined by the inner

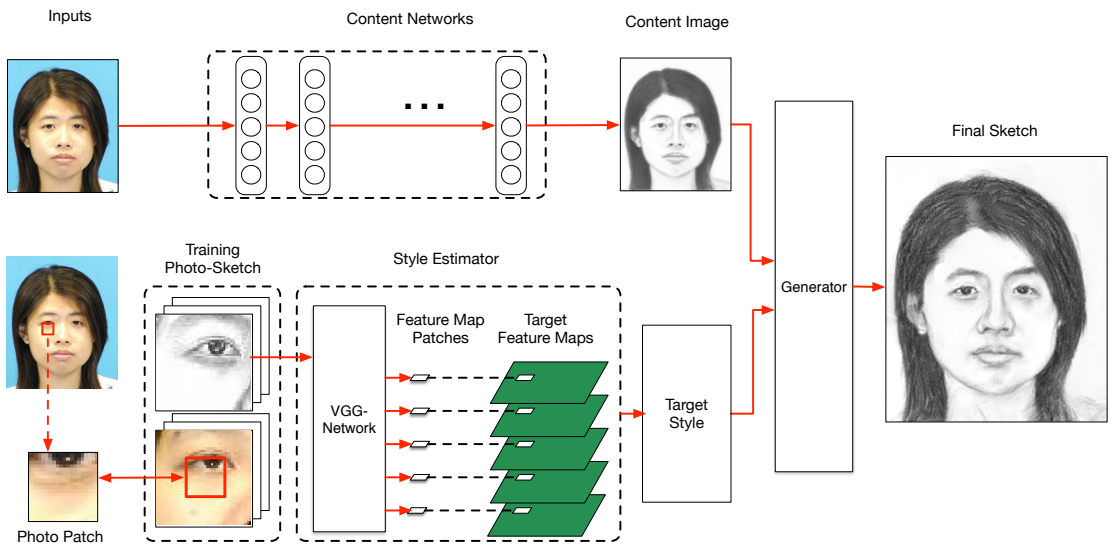


Figure 2. The proposed method contains two branches which take an eye-aligned test photo as input. The content network outputs a content image which outlines the face, and the style estimator generates a target style. The final sketch is generated by combing the target style with the content image.

products between two channels of this feature map, i.e.,

$$G^l_{ij}(\mathcal{X}) = F^l_i(\mathcal{X}) \cdot F^l_j(\mathcal{X}), \tag{1}$$

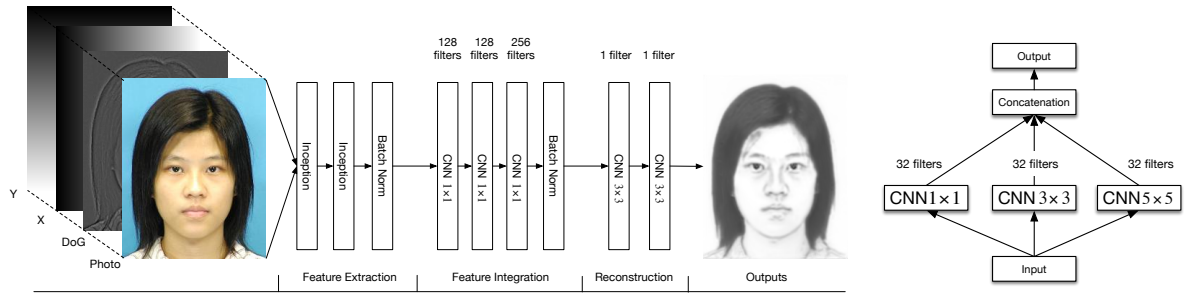
where $G^l(\mathcal{X}) \in \mathcal{R}^{N_l \times N_l}$ and N_l is the number of channels of the feature map in the lth layer. Since $G^l_{ij}(\mathcal{X})$ is an inner product between two channels of the feature map, a Gram matrix is actually a summary statistics of the feature map without any spatial information. Empirically , a Gram matrix of the feature map captures the density distribution of a sketch. For example, if a given style (sketch) image has much less hair than the test photo, the synthesized sketch $\mathcal{X}$ will become brighter than a natural sketch (see experimental results in Section 5.1). Thus it is important to have a style (sketch) image which is (statistically) similar to the test photo. Note that, in face sketch synthesis, however, there usually does not exist a single photo-sketch pair in the training set that matches all properties of the test photo. How to compute a target style for the synthesized sketch $\mathcal{X}$ is therefore not trivial, and is the key to the success of this approach. We will introduce a feature-space patch-based approach to solve this problem in Section 4.2.

4. Methodology

Our method can be classified as a shading synthesis method. The steps of our method are summarized in Fig. 2. First, a preprocessing step as described in [14] is carried out to align all photos and sketches in the training set by the centers of the two eyes. An eye-aligned test photo $\mathcal{I}$ is then fed into two branches, namely the content network and the style estimator. The content network converts $\mathcal{I}$ into a content image $\mathcal{C}$, which outlines the shape of the face and the key facial features such as nose, eyes, mouth and hair. The style estimator divides $\mathcal{I}$ into a grid of non-overlapping 16×16 patches. For each test patch, it locates the most similar photo patch from the photo-sketch pairs in the training set and produce a target sketch patch from the corresponding sketch in the pair. A pyramid column feature (Section 4.2) is then computed for the target sketch patch. Finally, a target style can be computed from a grid of these pyramid column features, and a final sketch $\mathcal{X}$ can be synthesized by applying the target style to $\mathcal{C}$ through neural style transfer [5].

4.1. Content Image Generation

The architecture of our content network is shown in Fig. 3. Besides the test photo, we feed three extra channels containing the spatial information (i.e., x and y coordinates) and a difference of Gaussian (DoG) image into our content network. As pointed out in [14], face sketch synthesis algorithms can benefit from integrating features from multiple resolutions. Hence, we employ an inception module [12] for feature extraction, which concatenates features generated from three groups of filters with a size of 1×1, 3×3 and 5×5 respectively (see Fig. 3(b)). Features extracted using a two-layer-inception module are then fed into a three-layer-CNN for feature integration, where all filters have a size of 1×1. Finally, the integrated features are used to reconstruct the content image $\mathcal{C}$ by a two-layer-CNN with the filter size being 3×3. Since L_1-norm is better at preserving details than L_2-norm, we use the L_1-norm between $\mathcal{C}$ and the ground truth sketch S as the loss function in train-



(a) The architecture of content network (b) Inception module

Figure 3. Illustration of the content network for generating a content image. The numbers above the building block denote the number of CNN filters. (a) The architecture of content network. (b) The inception module in (a) contains three groups of filters with different sizes.

ing our content network, i.e.,

$$\mathcal{L}_s = \frac{1}{N}\sum_{i=1}^{N} \|\mathcal{C}_i - S_i\| \tag{2}$$

where N is the number of training photos.

4.2. Style Estimation

As mentioned previously, there usually does not exist a single photo-sketch pair in the training set that matches all properties of the test photo. In order to estimate a target style for the final sketch $\mathcal{X}$, we subdivide the test photo into a grid of non-overlapping 16×16 patches. For each test patch, similar to previous work [14, 20], we find the best matching photo patch from the photo-sketch pairs in the training set in terms of Mean Square Error (MSE) *(ken: defines MSE)*. A target sketch patch can then be obtained from the corresponding sketch in the photo-sketch pair containing the best matching photo patch. Instead of compositing a style image using the thus obtained target sketch patches, which may show inconsistency across neighboring patches, we adopt a feature-space approach here. We extract feature patches from the feature maps of the original sketch at 5 different layers of the VGG-Network, namely $conv1\_1$, $conv2\_1$, $conv3\_1$, $conv4\_1$ and $conv5\_1$ respectively, that correspond to the target sketch patch. These feature patches have a size of 16×16, 8×8, 4×4, 2×2 and 1×1 respectively (see Fig. 4). We group these five feature patches of a target sketch patch and call it a *pyramid column feature*. Finally, a target style, in the form of Gram matrices, can be computed directly from a grid of such pyramid column features.

4.3. Loss Function for Sketch Generation

Similar to [5], our loss function is composed of a content loss and a style loss. In addition, we introduce a component loss to enhance the key facial components. The total loss is

$$\mathcal{L}_t(\mathcal{X}) = \alpha\mathcal{L}_c + \beta_1\mathcal{L}_s + \beta_2\mathcal{L}_k, \tag{3}$$

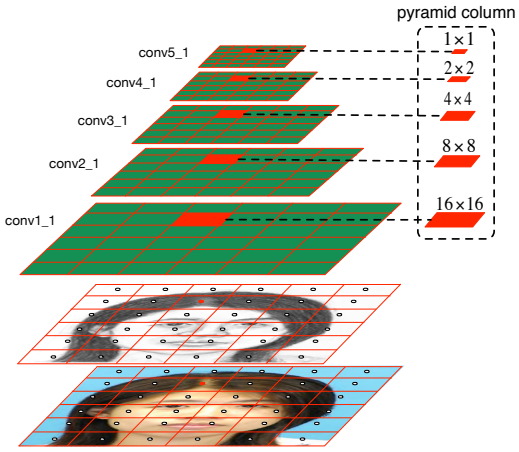


Figure 4. Illustration of the *pyramid column feature*. After finding a target sketch patch, we can extract its corresponding feature patches from the original feature maps. These five patches make up the *pyramid column feature*. *(ken: rewrite this caption)*

where α, β and β_2 are the weights for the different loss terms. We minimize the loss function by updating the target sketch $\mathcal{X}$.

The content loss is defined by the difference between the feature map at layer conv1_1 of the synthesized sketch and that of the content image :

$$\mathcal{L}_c(\mathcal{X}) = \|F^{\text{conv1\_1}}(\mathcal{X}) - F^{\text{conv1\_1}}(\mathcal{C})\|_2^2. \tag{4}$$

The style loss is defined by the difference between the Gram matrices of the synthesized sketch and that of the target style:

$$\mathcal{L}_s(\mathcal{X}) = \sum_{l \in L_s} \frac{1}{M_l^2 N_l^2} \|G^l(\mathcal{X}) - G^l(\mathcal{T})\|_2^2 \tag{5}$$

where N_l denotes the number of channels of the feature map at layer l, and M_l is the product of width and height of the feature map at layer l, and $G^l(\mathcal{T})$ is the Gram matrix computed from the grid of pyramid column features.

To better transfer styles of the key facial components, we employ a component loss to encourage the key component style of the final sketch to be the same as the target key component style. Since all photos and sketches have been aligned by the centers of the two eyes, the key components lie roughly within a rectangular region $\mathcal{R}$ with the eyes positioned at its upper corners. Here, we define the key component style by Gram matrices computed from feature maps corresponding to the rectangular region $\mathcal{R}$. The component loss is defined as

$$\mathcal{L}_k(\mathcal{X}) = \sum_{l \in L_s} \frac{1}{\hat{M}_l^2 N_l^2} \|\hat{G}^l(\mathcal{X}) - \hat{G}^l(\mathcal{T})\|_2^2 \qquad (6)$$

where $\hat{G}^l$ denotes the Gram matrix computed for the rectangular region $\mathcal{R}$, and $\hat{M}_l$ is the product of width and height of the feature map at layer l corresponding to $\mathcal{R}$.

4.4. Implementation Details

VGG-19 Parameters Since the VGG-Network is originally designed for color images, while sketches are gray scale images, we modify the first layer of VGG-Network for gray scale images by setting the filter weights to

$$W^k = W_r^k + W_g^k + W_b^k \qquad (7)$$

where W_r^k, W_g^k, and W_b^k are weights of the kth filter in the first convolutional layer for the R, G and B channels respectively, and W^k is the weight of the kth filter in the first convolutional layer of our modified network.

Data Partition CUHK [14] *(ken: ref?)* has 88 training photos and 100 test photos, and AR [9] *(ken: ref?)* has 123 photos. Our training set is composed of the 88 training photos of CUHK and 100 photos from AR. When training the content network, 10% of the training set are taken out as the validation set. All the 188 photo-sketch pairs are used to generate target sketch.

Training the Content Network The input photo-sketch pairs are all resized to 288×288 *(ken: not 288×288?)* and aligned by the centers of the two eyes. A mirror padding is carried out before the convolution operation when necessary to ensure the output sketch is of the same size as the input *(ken: ?)*. Adadelta [16] is used as the optimizer because it is stable and much faster than others.

Sketch Generation In all experiments, we resize the test photos and the photo-sketch pairs in the training set to 288×288. The final sketch is obtained by resizing the resulting sketch back to the original size. The size of $\mathcal{R}$ is 48×48. The weights in Eq. (3) are $\alpha = 0.004$, $\beta_1 = 1$ and $\beta_2 = 0.1$. The minimization is carried out using L-BFGS. Instead of using random noises, we use the content image as a starting point, which will make the optimization process converge much faster.

5. Experiments

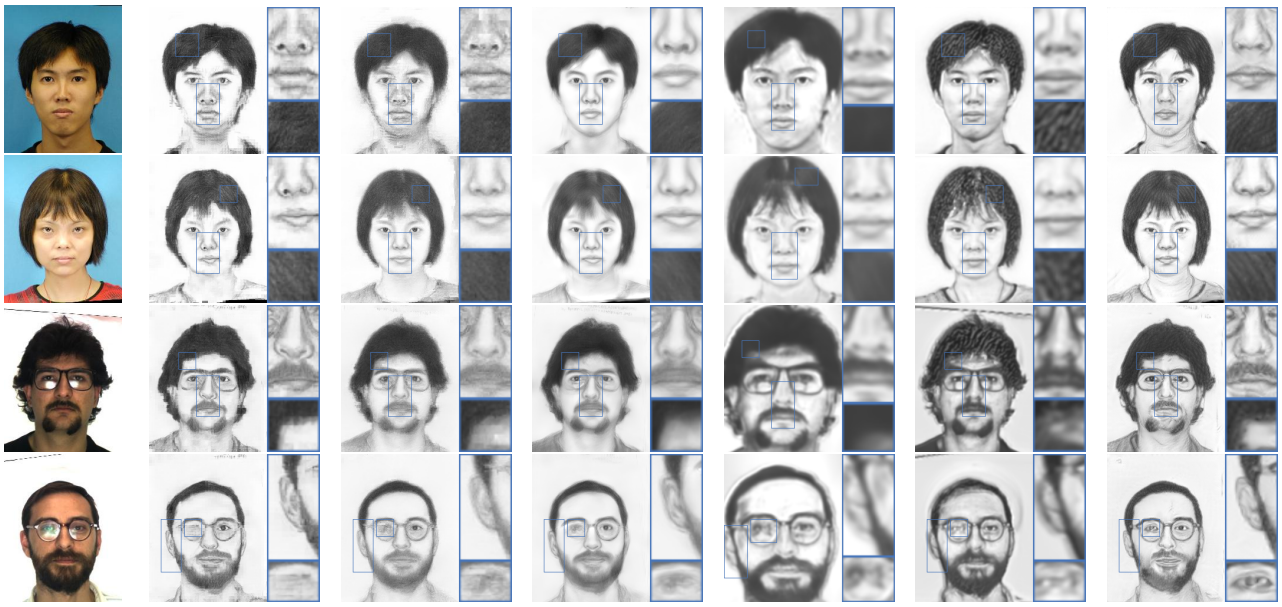
We evaluate the performance of the proposed method against other state-of-the-art methods on the CUHK student dataset [14] and the AR dataset [9]. We compare the results of our method against other 6 methods including traditional approaches and recent deep learning models. After discussing the disadvantages of previous quantitative evaluation criteria, we introduce the Normalized Gram Matrix Difference (NGMD) as a new evaluation tool. We believe that NGMD is more effective and will greatly promote the sketch qualities in future work.

5.1. Style Transfer Evaluation

(a) (b) (c) (d)

Figure 5. (a) is the content image generated by our content network. (b), (c) and (d) are generated by different styles. It can be seen that when the hair decreases the generated sketch becomes brighter. Results are generated by deepart website.

Although style transfer has shown remarkable performance in artistic style, it can't be directly applied to face sketch synthesis, see Fig. 1(h). The generated sketch is greatly influenced by illumination of original photo and doesn't look like sketch at all. To prove our assumption in Section 3 that gram matrix captures the density distribution of sketch, we replace the original RGB photo with content image generated by our content network. We selected 3 sketch styles with different amount of hairs and see how it influences the result. Fig. 5 shows a clear relationship between the hair amount of sketch and pixel intensities of sketch result. The facial key parts in Fig. 5(c) and 5(d) are missed. Therefore, even with a content structure, the original style transfer is still not suitable for elaborate task. However, if the style image has a similar structure with test photo, for example Fig. 5(b), the results can be quite good. This inspires our feature patch based method. Image patch method is not considered because it will introduce patch inconsistency as discussed in Section 2.



(a) Photo (b) MRF (c) WMRF (d) SSD (e) FCNN (f) BFCN (g) Ours

Figure 6. Examples of qualitative evaluation on CUHK (first two rows) and AR (last two rows). (a) The sketches drawn by artists. (b) MRF [14] (c) WMRF [20] (d) SSD [11] (e) FCNN [18] (f) BFCN [17] (g) Ours. The proposed method preserves more texture details, for example in the hair and nose. It is also best at keeping the origin structures of photos, such as the glasses.

5.2. Sketch Generation

Fig. 6 shows the comparison between our methods and many other approaches. The first two rows are from CUHK test set, and the last two are from AR. We can see that our method can generate more stylistic sketches than others. For example, in the hair part, only MRF, BFCN and the proposed method can generate obvious textures. However, the texture of MRF is not continuous in the border part and introduce many other artifacts, and the texture of BFCN doesn't look like human strokes. Both WMRF and SSD introduce over smoothing effect and FCNN is not able to give clear textures. Our method can not only generate textures for hairs and mustache but also shadings, for example the nose part.

On the other hand, only BFCN and the proposed method can handle structures decorated on the face well, for example the glasses of the last two row. MRF, WMRF and SSD are exemplar based method so they can't handle something different from training set, the glass edges of them are not complete. FCNN, BFCN and our method generate the image content by CNN, so they can handle the original photo structures well. But both FCNN and BFCN care nothing about the textures of facial part, so their results are not good considering the face part. In contrast, our method can well maintain the image content and attach textures similar to human paintings.

Methods	AR			CUHK		
	R1	R5	R10	R1	R5	R10
FCNN	-	-	-	81%	96%	97%
MRF	97.5%	97.5%	100%	83%	96%	96%
WMRF	97.5%	97.5%	100%	83%	97%	98%
SSD	96.7%	97.5%	100%	87%	97%	98%
Ours	98.4%	98.4%	100%	87%	98%	99%

Table 1. Recognition rate on benchmark datasets. The best performance is colored in red.

5.3. Quantitative Results

Sketch Recognition Sketch synthesis methods are usually evaluated quantitatively via the face sketch recognition task [11, 14, 18, 20]. If an algorithm achieves higher sketch recognition rates, it suggests that this method is more effective in synthesizing sketches. We adopt the widely used PCA based recognition method with "rank-1 (R1)", "rank-5 (R5)" and "rank-10 (R10)" criteria [14] where "rank n" measures the rate of the correct answer in the top n best matches. The results of different methods are shown in Table 1. Our method achieves the best performance against all other methods in the "R1" and "R5" tests.

However, such kind of evaluation is not effective since every one is close to 100%, and more importantly, it can't guide our research effort. Zhang *et al*. [18] proposed a Multiscale Pixel-wise Reconstruction Loss (MPRL). Although we agree that if MPRL is zero, the generated sketches will be exactly the same with ground truth, there are two rea-

Methods	AR					CUHK				
	conv1_1	conv2_1	conv3_1	conv4_1	conv5_1	conv1_1	conv2_1	conv3_1	conv4_1	conv5_1
FCNN	-	-	-	-	-	0.009	0.110	0.080	9.43	1.49
MRF	0.0043	0.009	0.033	0.12	0.28	0.010	0.014	0.047	0.13	0.18
WMRF	0.0053	0.027	0.085	0.19	0.29	0.010	0.052	0.052	0.27	0.19
SSD	0.0056	0.036	0.110	1.90	0.28	0.009	0.102	0.070	3.32	0.24
Ours	0.0035	0.008	0.029	0.08	0.17	0.007	0.012	0.033	0.07	0.12

Table 2. Averaged NGMD value of different methods at different level on AR and CUHK datasets. A smaller NGMD indicates the generated texture is more similar to ground truth.

sons why it is not a good criteria. First, experiment results of [18, 17] show that L_2-norm would give blurry sketches. Second, the ground truth draw by human usually doesn't correspond exactly with photos, thus difficult for algorithms to reach the ground truth.

Meanwhile, human can often be able to tell whether a sketch is generated by algorithms or drawn by artists just from a quick glance without examining details. This indicates that there exists a big gap between the style of algorithm-generated sketches and that of those drawn by artists. And we need a criteria which can quantitatively measure such gap.

Normalized Gram Matrix Difference (NGMD) To quantitatively evaluate the style similarity between a generated sketch and the sketch drawn by a real artist, we employ the normalized gram matrix difference (NGMD) between the generated sketch $\mathcal{X}$ and the drawn sketch $\tilde{\mathcal{X}}$:

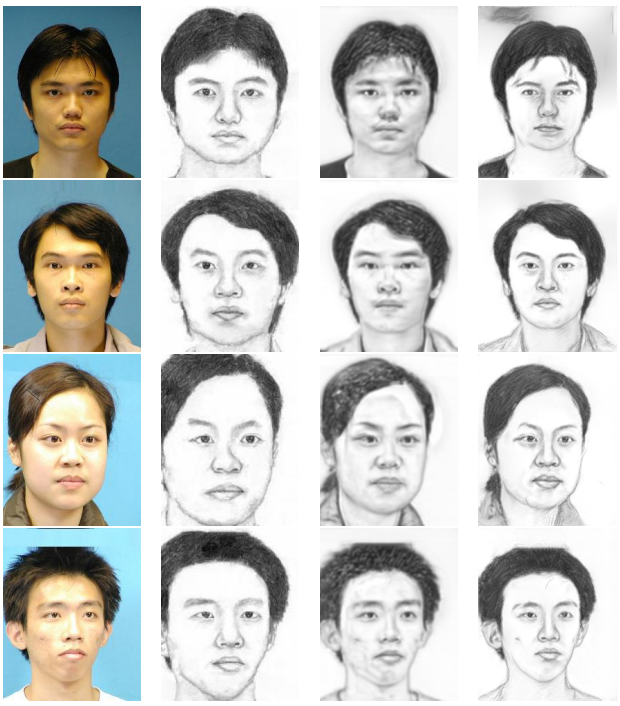
$$D_s^l\left(\mathcal{X},\tilde{\mathcal{X}}\right)=\frac{\left\|G^l\left(\mathcal{X}\right)-G^l\left(\tilde{\mathcal{X}}\right)\right\|_2^2}{\left\|G^l\left(\tilde{\mathcal{X}}\right)\right\|_2^2} \tag{8}$$

where $l \in L_s$. The smaller the NGMD value is, the more similar $\mathcal{X}$ and $\tilde{\mathcal{X}}$ are.

It is exciting how well NGMD match our intuition, see Fig. 6. The results of [18] can roughly outline the sketches but can't add textures, thus they got a high NGMD in Table 2. The results of MRF [14] and WMRF [20] are more similar to hand drawn sketches in style since these exemplar based approaches use patches of hand drawn sketches to make up the target sketch. That is why they have a smaller NGMD value. Inheriting the ability of denoising, the filtering based approach [11] is good at suppressing noises in the results. However, it is also likely for this method to oversmooth the results, which will deteriorate the texture, thus its NGMD value is higher than WMRF.

5.4. Generalization Evaluation

In this part, we will evaluate the generalization ability of our model in two aspects.

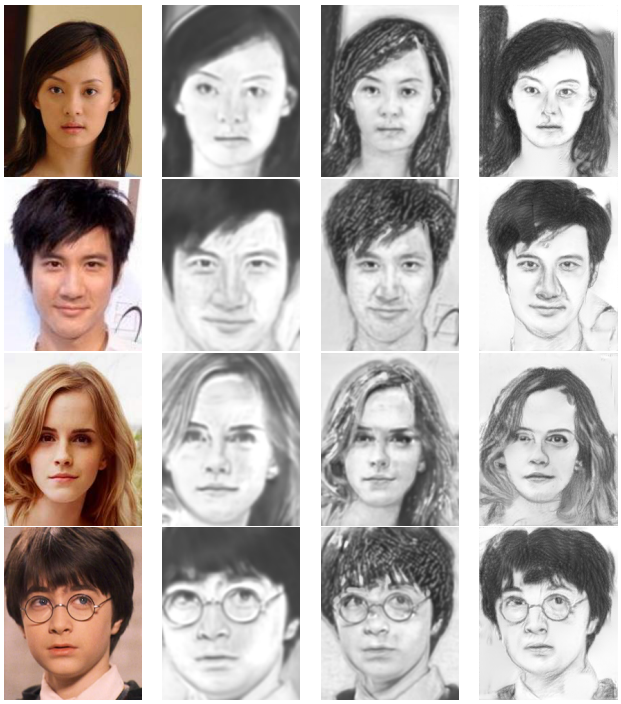


(a) Photo (b) MRF(extend) (c) BFCN (d) Ours

Figure 7. Experiment with different light and pose. Our results are little affected by light and pose change.

Light and Pose Invariance As discussed in [19], light and pose change may influence the result a lot. We choose several photos from [19] and compare our results with MRF(extended) [19] and BFCN [17]. Fig. 7 shows the comparison. Our proposed method is not influenced by pose and light change and can still generate good textures under lab environment.

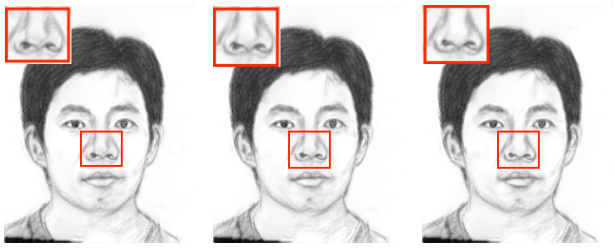
Real World Photos We further tested the robustness of our model on some real world photos, see Fig. 8. The first two rows are Chinese celebrity faces from [19], and the latter two comes from the web. Since the test photo may not be well aligned, we just turn off the component loss. The parameters we use here are $\alpha = 0.004$, $\beta_1 = 1$, $\beta_2 = 0$. Although the background is clutter and the positions of faces



(a) Photo (b) FCNN (c) BFCN (d) Ours

Figure 8. Experiments with real world photos. Our method can still get better results under practical situations.

are not strictly constrained. The hair style of our results are still clear and sharp, while FCNN and BFCN can't produce good textures.



(a) (a) $\beta_2 = 0.001$ (b) (b) $\beta_2 = 0.01$ (c) (b) $\beta_2 = 0.1$

Figure 9. Comparison between results with different weight of $\mathcal{L}_k$ regulation. With the increase of β_2 the distortion of nose becomes less.

5.5. Effectiveness of the model

The loss function we minimize during the generation of sketches contains three terms for content, style and key components respectively. The term $\mathcal{L}_k$ regularizes the results by encouraging the style extracted from the key component regions in the training set to be placed into the key components region of the results, which helps generate better results around these components (see Fig. 9). To better understand how style influences the final sketch, we smoothly change the emphasis on style by adjusting β_1 and β_2 while keeping α fixed. Fig. 10 indicates that the sketch with style transfered contains more texture and is more like a drawn sketch. The Theano implementation of the proposed method takes approximately 100 seconds to generate a sketch on a GeForce GTX TITAN X platform. The bottle neck lies in the style transfer which requires feeding $\mathcal{X}$ to the VGG-Network to estimate targeting feature maps and to calculate the gradient of Eq. (3), which is computationally intensive.

6. Conclusion

This paper proposed a novel face sketch synthesis method inspired by the procedure of artists drawing sketches. In our method, the outline of the face is delineated by a content network and the style extracted from sketches drawn by artists are transferred to generate a final sketch. Quantitative evaluations on face sketch recognition and style similarity measure demonstrate the effectiveness of the proposed algorithm for face sketch synthesis and style transferring. Our future work will investigate accelerating technique to reduce the running time and achieve real time face sketch synthesis with style transfer.

References

[1] I. Berger, A. Shamir, M. Mahler, E. Carter, and J. Hodgins. Style and abstraction in portrait sketching. *ACM Transactions on Graphics (TOG)*, 32(4):55, 2013. 2

[2] H. Chen, Y.-Q. Xu, H.-Y. Shum, S.-C. Zhu, and N.-N. Zheng. Example-based facial sketch generation with non-parametric sampling. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 433–438. IEEE, 2001. 2

[3] T. Q. Chen and M. Schmidt. Fast patch-based style transfer of arbitrary style. *CoRR*, abs/1612.04337, 2016. 2

[4] L. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 262–270, 2015. 2

[5] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015. 1, 2, 3, 4

[6] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. 2

[7] J. Justin, A. Alexandre, and F.-F. Li. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711, 2016. 2

[8] Q. Liu, X. Tang, H. Jin, H. Lu, and S. Ma. A nonlinear approach for face sketch synthesis and recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 1005–1010. IEEE, 2005. 2

[9] A. Martinez. R. benavente. the AR face database. Technical report, CVC Tech. Report, 1998. 5

$\beta_1 = 0$	$\beta_1 = 10^{-3}$	$\beta_1 = 10^{-2}$	$\beta_1 = 0.1$	$\beta_1 = 1$	$\beta_1 = 10^3$
$\beta_2 = 0$	$\beta_2 = 10^{-4}$	$\beta_2 = 10^{-3}$	$\beta_2 = 0.01$	$\beta_2 = 0.1$	$\beta_2 = 10^2$

Figure 10. Final results for sketch synthesis with fixed $\alpha = 0.004$ and different β_1, β_2 values.

[10] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

[11] Y. Song, L. Bao, Q. Yang, and M.-H. Yang. Real-time exemplar-based face sketch synthesis. In *ECCV*, pages 800–813, 2014. 1, 2, 6, 7

[12] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015. 2, 3

[13] X. Tang and X. Wang. Face sketch synthesis and recognition. In *IEEE International Conference on Computer Vision*, pages 687–694. IEEE, 2003. 2

[14] X. Wang and X. Tang. Face photo-sketch synthesis and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(11):1955–1967, 2009. 1, 2, 3, 4, 5, 6, 7

[15] Z. Xu, H. Chen, S.-C. Zhu, and J. Luo. A hierarchical compositional model for face representation and sketching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(6):955–969, 2008. 2

[16] M. D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012. 5

[17] D. Zhang, L. Lin, T. Chen, X. Wu, W. Tan, and E. Izquierdo. Content-adaptive sketch portrait generation by decompositional representation learning. *IEEE Transactions on Image Processing*, 26(1):328–339, 2017. 1, 2, 6, 7

[18] L. Zhang, L. Lin, X. Wu, S. Ding, and L. Zhang. End-to-end photo-sketch generation via fully convolutional representation learning. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, pages 627–634. ACM, 2015. 1, 2, 6, 7

[19] W. Zhang, X. Wang, and X. Tang. Lighting and pose robust face sketch synthesis. In *Computer Vision–ECCV 2010*, pages 420–433. Springer, 2010. 1, 2, 7

[20] H. Zhou, Z. Kuang, and K.-Y. K. Wong. Markov weight fields for face sketch synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1091–1097. IEEE, 2012. 1, 2, 4, 6, 7