# Face Sketch Synthesis with Style Transfer using Pyramid Column Feature

Chaofeng Chen[1]*, Xiao Tan[2*†], and Kwan-Yee K. Wong[1]

[1]The University of Hong Kong, [2]Baidu Research

{cfchen, kykwong}@cs.hku.hk, tanxchong@gmail.com

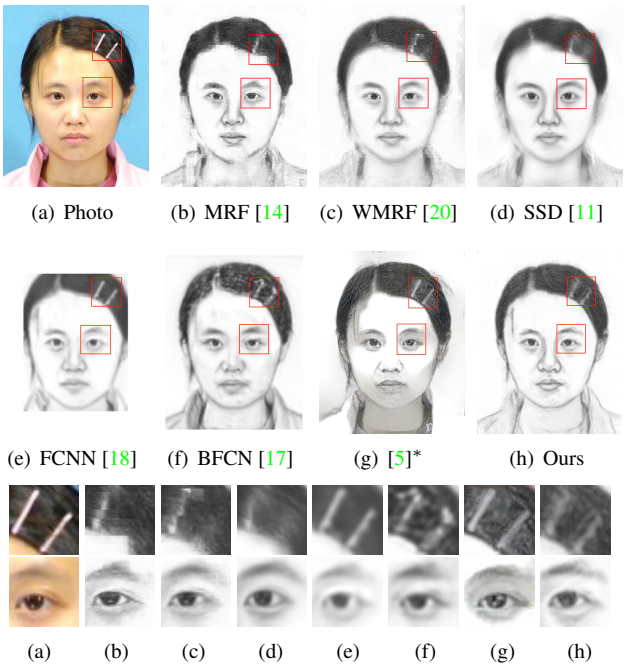
## Abstract

*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 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.



(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. For results of different sizes, we simply padded them with zeros.)

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

*indicates equal contribution

†This work was done when Xiao Tan was a postdoc at HKU.

[1]https://deepart.io/

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 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 brightness of the result is easily influenced by the content of the style image (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)).

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 [5], 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 preserves the content of the test photo better than current state-of-the-art methods.
- It achieves better results in the face sketch recognition task.

## 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 methods 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 not 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 tar-

get 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 $c$th channel of the feature map in the $l$th layer of the final sketch $\mathcal{X}$ by $F_c^l(\mathcal{X})$. A Gram matrix of the feature map in the $l$th layer is then defined by the inner products between two channels of this feature map, i.e.,

$$G_{ij}^l(\mathcal{X}) = F_i^l(\mathcal{X}) \cdot F_j^l(\mathcal{X}), \tag{1}$$

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 $l$th layer. Since $G_{ij}^l(\mathcal{X})$ 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 5.3). 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 $16 \times 16$ 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 $1 \times 1$, $3 \times 3$ and $5 \times 5$ 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 $1 \times 1$. Finally, the integrated features are used to reconstruct the content image $\mathcal{C}$ by a two-layer-CNN with the filter size being $3 \times 3$. 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 training our content network, i.e.,

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

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 $16 \times 16$ 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). 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 $16 \times 16$, $8 \times 8$, $4 \times 4$, $2 \times 2$ and $1 \times 1$ 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

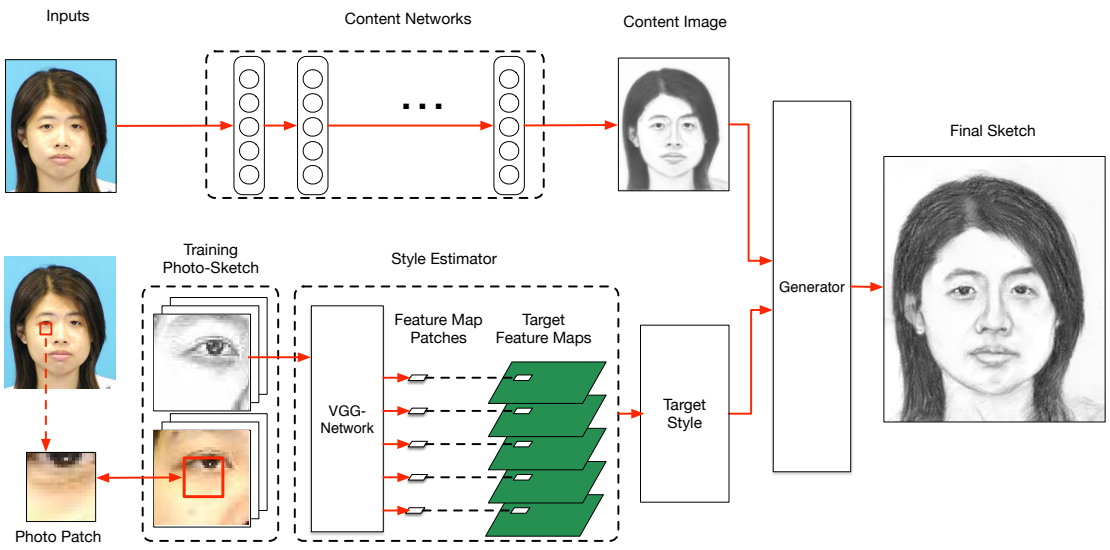
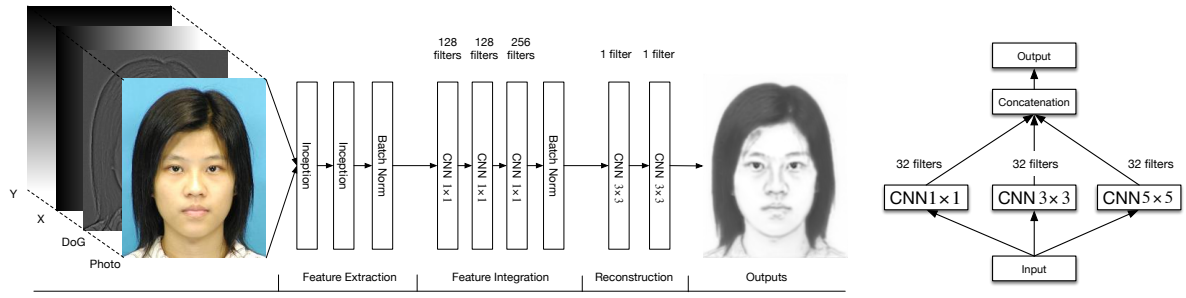



Figure 2. The proposed method contains two branches which take an eye-aligned test photo as input. The content network outputs a content image which outlines the face, and the style estimator generates a target style. The final sketch is generated by combing the target style with the content image.




(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.

loss to enhance the key facial components. The total loss is

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

where $\alpha$, $\beta_1$ and $\beta_2$ are the weights for the different loss terms. We minimize the loss function by updating the target sketch $\mathcal{X}$ in the same way as [5].

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:

$$\mathcal{L}_s(\mathcal{X}) = \sum_{l\in L_s} \frac{1}{4M_l^2N_l^2}\|G^l(\mathcal{X}) - G^l(\mathcal{T})\|_2^2 \tag{5}$$

where $N_l$ denotes the feature map channels 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

$$\mathcal{L}_k(\mathcal{X}) = \sum_{l\in L_s} \frac{1}{4\hat{M}_l^2N_l^2}\|\hat{G}^l(\mathcal{X}) - \hat{G}^l(\mathcal{T})\|_2^2 \tag{6}$$

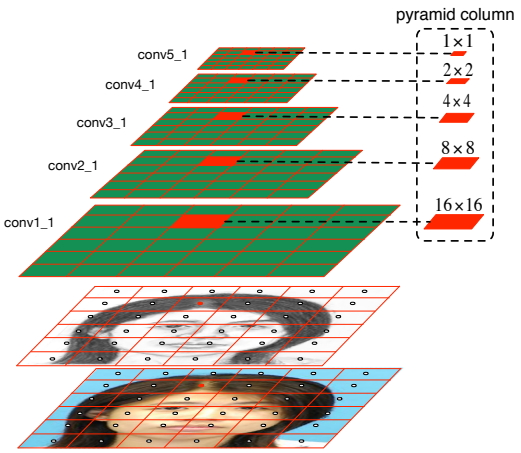



Figure 4. Illustration of the *pyramid column feature*. After finding a target sketch patch, we can extract its corresponding feature patches from the original feature maps. These five feature patches make up the *pyramid column feature.*

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

$$W^k = W_r^k + W_g^k + W_b^k \tag{7}$$

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

**Data Partition** CUHK [14] has 88 training photos and 100 test photos, and AR [9] has 123 photos. Our training set is composed of the 88 training photos from 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 $288 \times 288$ and aligned by the centers of the two eyes. A mirror padding is carried out before the convolution operation except when kernel size is $1 \times 1$ to ensure the output sketch is of the same size as the input. 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 $288 \times 288$. The final sketch is obtained by resizing the resulting sketch back to the original size. The size of $\mathcal{R}$ is $48 \times 48$. 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 five other methods, including traditional approaches and recent deep learning models. After discussing the reason why direct style transfer fails, we evaluate the effectiveness of the proposed *pyramid column feature* and different components of our loss function.

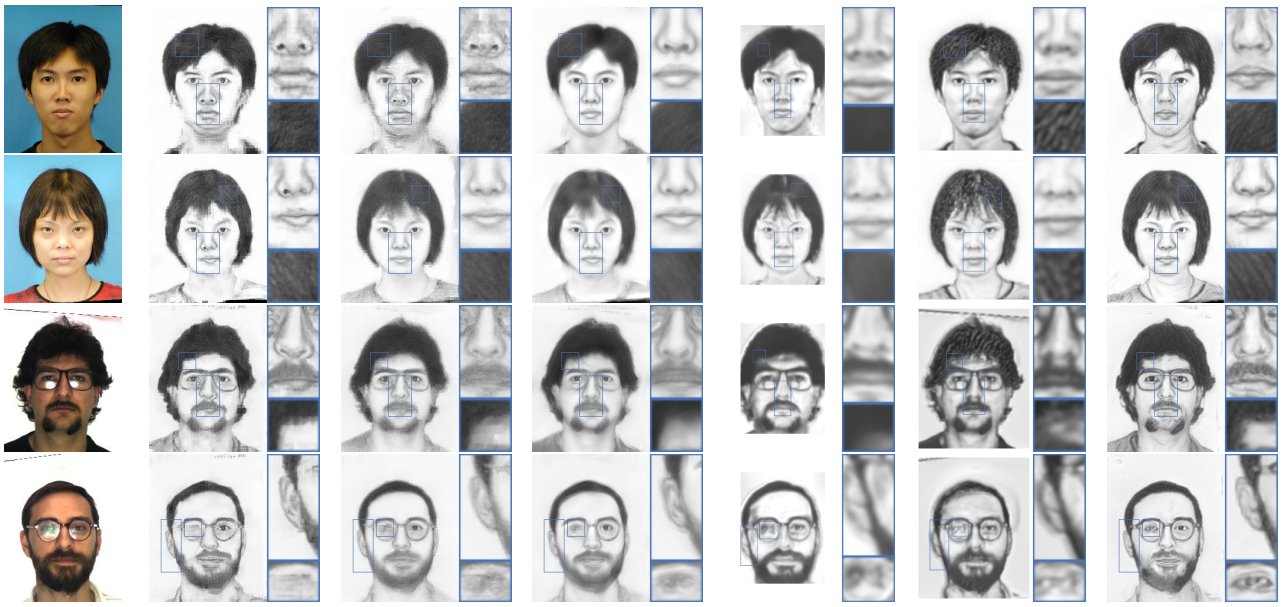
### 5.1. Qualitative Evaluation

Fig. 5 shows the comparison between our methods and five other methods. The first two rows are from the CUHK dataset, and the last two are from the AR dataset. We can see that our method can generate more stylistic sketches than the others. For example, in the hair part, only MRF, BFCN and the proposed method can generate obvious textures. However, the texture in the results of MRF is not continuous across image patches and shows many unpleasant artifacts, whereas the texture in the results of BFCN does not look like natural hand strokes. Both WMRF and SSD introduce an over smoothing effect, and FCNN is not able to produce clear texture. Our method can not only generate textures for hair and moustache, but also shadings (e.g., around the nose).

On the other hand, only FCNN, BFCN and the proposed method can handle structures decorated on the face well, for example the glasses in the last two rows. MRF, WMRF and SSD are exemplar based methods and therefore they cannot handle structures different from the training set. The edges of glasses are not complete in their results. FCNN, BFCN and our method generate the image content by CNN, so they can handle the original structures in the test photo well. However, both FCNN and BFCN cannot generate sharp edges. For example, they produce results which are over smooth in the facial regions around the nose and mouth (see Fig. 5). In contrast, our method can well maintain the image content and create sharp edges.

### 5.2. Quantitative Evaluation

Sketch synthesis methods are commonly evaluated quantitatively via the face sketch recognition task [11, 14, 18, 20]. If an algorithm achieves higher sketch recognition



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

Figure 5. Qualitative evaluation of the face sketch synthesis results on the CUHK (first two rows) and AR (last two rows) datasets. (a) Original Photos. (b) MRF [14] (c) WMRF [20] (d) SSD [11] (e) FCNN [18] (f) BFCN [17] (g) Ours. The proposed method preserves more texture details (e.g., in the hair and nose), and is also best at keeping the origin structures (e.g., glasses) in the photos.

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

| Methods | AR | | | CUHK | | |
|---|---|---|---|---|---|---|
| | R1 | R5 | R10 | R1 | R5 | R10 |
| 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% |
| FCNN | - | - | - | 81% | 96% | 97% |
| BFCN | 92.7% | 100% | 100% | 83% | 89% | 92% |
| Ours | 98.4% | 98.4% | 100% | 87% | 98% | 99% |

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 having 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 in all the "R1" and "R5" and "R10" tests.

### 5.3. Direct Style Transfer

Although style transfer has shown remarkable performance in transferring artistic style, it cannot be directly applied to face sketch synthesis as the brightness of the generated sketch is easily influenced by the content of the style image (see Fig. 1(g)). To demonstrate that the Gram matrix captures the density distribution of a sketch, we select 3 sketches with different amount of hair and directly transfer their styles to a content image generated by our content network. The results are shown in Fig. 6. We can see a clear relationship between the amount of hair in the style image and the pixel intensities of the generated sketch. Some key facial details in Fig. 6(c) and 6(d) are missing due to the overall elevation of pixel intensities. Obviously, direct application of style transfer with an arbitrary style image cannot produce a satisfactory face sketch result. On the other hand, a good looking sketch can be obtained if the style image has a structure similar to the test photo (see Fig. 6(b)). This inspires us to introduce our feature-space patch-based method in Section 4.2 for generating a compatible target style for our content image.

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

Figure 6. (a) is the content image generated by our content network. (b), (c) and (d) are generated by different styles. It can be seen that the generated sketch becomes brighter as the amount of hair decreases in the style image.

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

Figure 7. Final results for sketch synthesis with fixed $\alpha = 0.004$ and different $\beta_1$ and $\beta_2$ values.

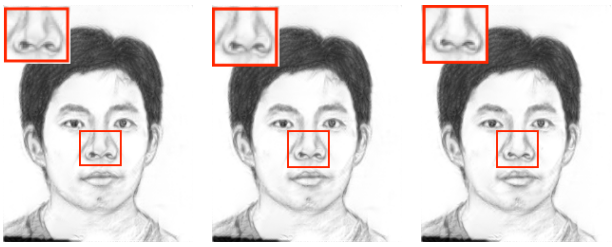
(a) (b) (c) (d)

Figure 8. Feature level style composition helps to eliminate blocky appearance. (a) Photo. (b) A style image composed from target sketch patches. (c) Face sketch obtained from the deep art website using (b) as the style. (d) Face sketch generated using pyramid column feature.

### 5.4. Effectiveness of the model

**Pyramid Column Feature** As demonstrated in the previous section, we need a style image with similar content to the content image in order to apply style transfer for face sketch synthesis. Here, we circumvent this limitation by aggregating pyramid column features of the target sketch patches in estimating the target style. Such a feature space composition is the key to success in our approach. As shown in Fig. 8, if we simply assemble a target style image from the sketch patches and apply style transfer naïvely, the blocky appearance will also be transferred. Although Fig. 8(b) can be improved by making the patches overlap and adding a smoothing term as in [14, 20], a simpler and more effective way is to compose the target style directly in the feature space, i.e., the proposed *pyramid column feature*. Since the receptive field of the CNN features grow exponentially, the receptive fields of high level feature patches overlap with each other, and this helps to eliminate the blocky appearance problem (see Fig. 8(d)).

**Loss Components** 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 key component style of the final sketch to be the same as the target key component style. This helps generate better results around the key facial components (see Fig. 9). To better understand how style influences the final sketch, we smoothly change the emphasis on style by adjusting $\beta_1$ and $\beta_2$ while keeping $\alpha$ fixed. Fig. 7 shows that the sketch with style transferred contains more textures and looks more like a drawn sketch.



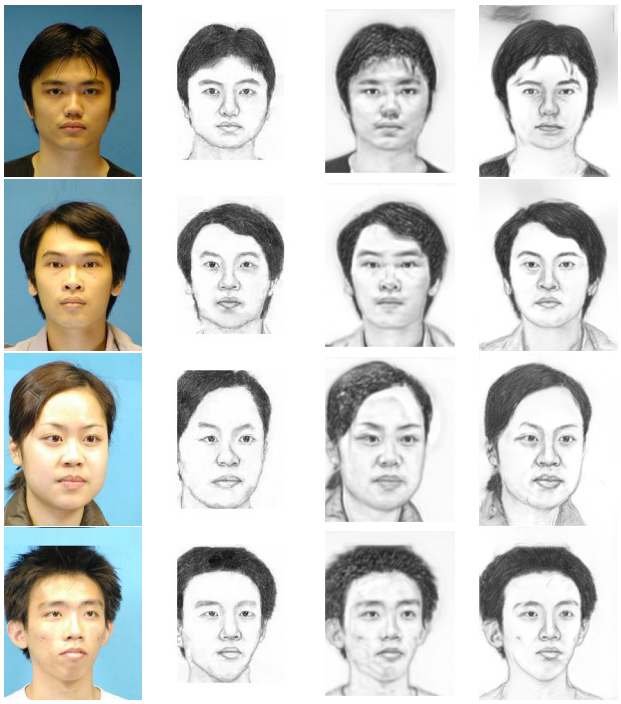
(a) $\beta_2 = 0.001$ (b) $\beta_2 = 0.01$ (c) $\beta_2 = 0.1$

Figure 9. Comparison between results with different weight of $\mathcal{L}_k$ regulation. With the increase of $\beta_2$ the distortion of nose becomes less.

### 5.5. Generalization

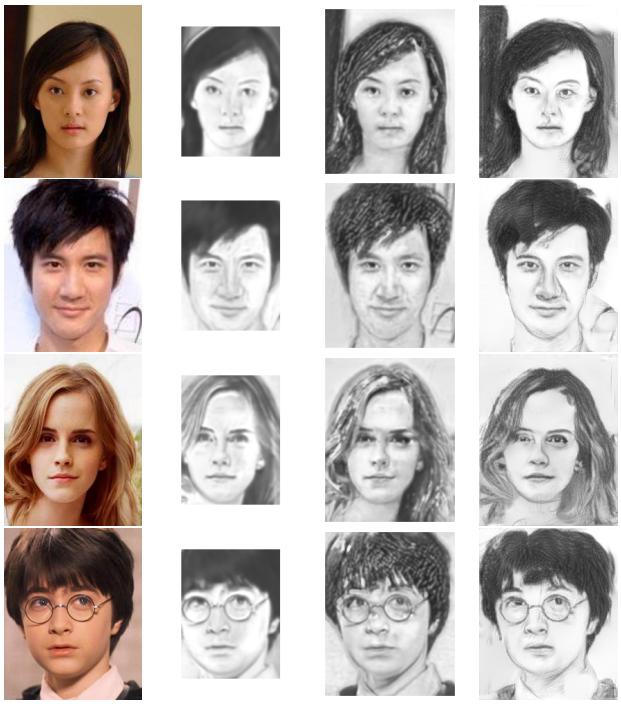
**Light and Pose Invariance** As discussed in [19], light and pose changes may influence the result a lot. We choose several photos from [19] and compare our results with MRF (extended) [19] and BFCN [17]. As seen from the comparison in Fig. 10, our proposed method is not influenced by pose and light changes, and can generate good textures under a laboratory environment.

**Real World Photos** We further test the robustness of our model on some real world photos, and the results are shown in Fig. 11. The first two rows are Chinese celebrity faces from [19], and the latter two come from the world wide web. Since the test photos 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 cluttered and the positions of the faces are not strictly constrained, the hair styles of our results are still clear and sharp, whereas FCNN and BFCN fail to produce good textures.



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

Figure 10. Experiment under different light and pose settings.



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

Figure 11. Experiments with real world photos.

## 6. Discussion

The proposed method has two key parts, content image generation and patch matching based sketch feature composition. When the content image is of low quality, the key facial parts of the result may be also unclear, such as the first row in Fig. 11. We believe that training the content network with more data will further improve the result. As for the patch matching part, one concern is that the database may not be suitable for all faces. Nevertheless, our database which contains 188 face sketch pairs can already handle most situations because of the limitation of face content. For example, different shape of glasses are well preserved in Fig. 5. Due to the iterative optimization process, the time complexity is another limitation of our method. The Theano implementation of the proposed method takes approximately 60 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 the feature maps and calculate the gradient of Eq. (3), which is computationally intensive.

## 7. Conclusion

This paper proposes a novel face sketch synthesis method inspired by the process of how artists draw 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 demonstrate the effectiveness of the proposed algorithm for face sketch synthesis. Our future work will investigate accelerating technique to reduce the running time and achieve real time face sketch synthesis with style transfer.

## 8. Acknowledgment

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU used for this research.

## 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 *IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 433–438, 2001. 2

[3] T. Q. Chen and M. Schmidt. Fast patch-based style transfer of arbitrary style. *arXiv preprint arXiv: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 (NIPS)*, 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. *International Conference on Machine Learning (ICML)*, pages 448–456, 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 (ECCV)*, 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 Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 1005–1010, 2005. 2

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

[10] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3

[11] Y. Song, L. Bao, Q. Yang, and M.-H. Yang. Real-time exemplar-based face sketch synthesis. In *European Conference on Computer Vision (ECCV)*, pages 800–813, 2014. 1, 2, 5, 6

[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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015. 2, 3

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

[14] X. Wang and X. Tang. Face photo-sketch synthesis and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(11):1955–1967, 2009. 1, 2, 3, 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 (PAMI)*, 30(6):955–969, 2008. 2

[16] M. D. Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv: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 (TIP)*, 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 (ICMR)*, pages 627–634, 2015. 1, 2, 5, 6

[19] W. Zhang, X. Wang, and X. Tang. Lighting and pose robust face sketch synthesis. In *European Conference on Computer Vision (ECCV)*, pages 420–433. 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 (CVPR)*, pages 1091–1097, 2012. 1, 2, 3, 5, 6, 7