

000  
001  
002003 

# Face Sketch Synthesis with Style Transfer using Pyramid Column Feature

004  
005  
006  
007  
008  
009  
010  
011

Anonymous ICCV submission

Paper ID 612

012  
013

## Abstract

014  
015  
016  
017  
018  
019  
020  
021  
022  
023  
024  
025  
026  
027  
028  
029  
030  
031

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.

032  
033  
034

## 1. Introduction

035  
036  
037  
038  
039  
040  
041  
042  
043  
044

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.

045  
046  
047  
048  
049  
050  
051  
052  
053

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

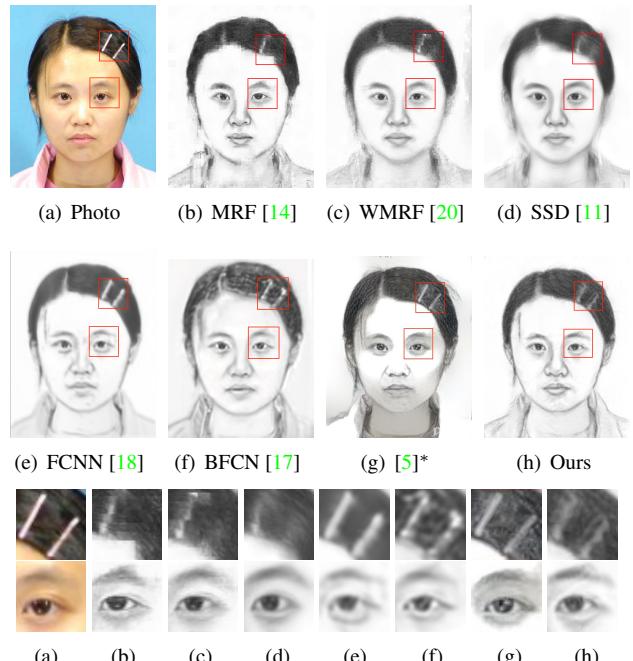
054  
055  
056  
057  
058  
059  
060  
061  
062  
063  
064  
065  
066  
067  
068  
069  
070  
071  
072  
073  
074  
075  
076  
077  
078  
079  
080  
081  
082  
083  
084  
085  
086  
087  
088  
089  
090  
091  
092  
093  
094  
095  
096  
097  
098  
099  
100  
101  
102  
103  
104  
105  
106  
107

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<sup>1</sup> using the photo as content and a sketch from the training set as style.)

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

<sup>1</sup><https://deeprart.io/>

108 by these methods will lead to blurry artifacts (see Fig. 1(e)  
109 and 1(f)) because they are incapable of preserving texture  
110 structures. The popular neural style transfer provides a better  
111 solution for texture synthesis. However, there exist two  
112 obstacles in directly applying such a technique. First, the  
113 brightness of the result is easily influenced by content of  
114 the style image (see the face in Fig. 1(g)). Second, it re-  
115 quires a style image to provide the global statistics of the  
116 textures. If the given style image does not match with the  
117 target sketch (which we do not have), some side effects will  
118 occur (see the nose in Fig. 1(g)).

119 For an artist, the process of sketching a face usually starts  
120 with outlining the shape of the face and the key facial fea-  
121 tures like the nose, eyes, mouth and hair. Textures and shad-  
122 ings are then added to regions such as hair, lips, and bridge  
123 of the nose to give the sketch a specific style. Based on  
124 the above observation, and inspired by neural style transfer  
125 [5], we propose a new framework for face sketch synthe-  
126 sis from a photo that overcomes the aforementioned limita-  
127 tions. In our method, a content image that outlines the face  
128 is generated by a feed-forward neural network, and textures  
129 and shadings are then added using a style transfer approach.  
130 Specifically, we design a new architecture of fully convolu-  
131 tional neural network (FCNN) composed of inception lay-  
132 ers [12] and convolution layers with batch normalization [6]  
133 to generate the content image (see Section 4.1). To synthe-  
134 size the textures, we first divide the target sketch into a grid.  
135 For each grid cell, we compute a newly proposed pyramid  
136 column feature using the training set (see Section 4.2). A  
137 target style can then be computed from a grid of these pyra-  
138 mid column features, and applied to the content image. Our  
139 approach is superior to the current state-of-the-art methods  
140 in that

- 142 • It is capable of generating more stylistic sketches with-  
143 out introducing over smoothing artifacts.
- 144 • It can preserve the content of the test photo well.

## 147 2. Related Work

### 148 2.1. Face Sketch Synthesis

149 Based on the taxonomy of previous studies [11, 20],  
150 face sketch synthesis methods can be roughly categorized  
151 into profile sketch synthesis methods [1, 2, 15] and shading  
152 sketch synthesis methods [8, 11, 13, 14, 18, 19, 20].  
153 Compared with profile sketches, shading sketches are more  
154 expressive and thus more preferable in practice. Based on  
155 the assumption that there exists a linear transformation be-  
156 tween a face photo and a face sketch, the method in [13]  
157 computes a global eigen-transformation for synthesizing a  
158 face sketch from a photo. This assumption, however, does  
159 not always hold since the modality of face photos and that  
160 of face sketches are quite different. Liu et al. [8] pointed out

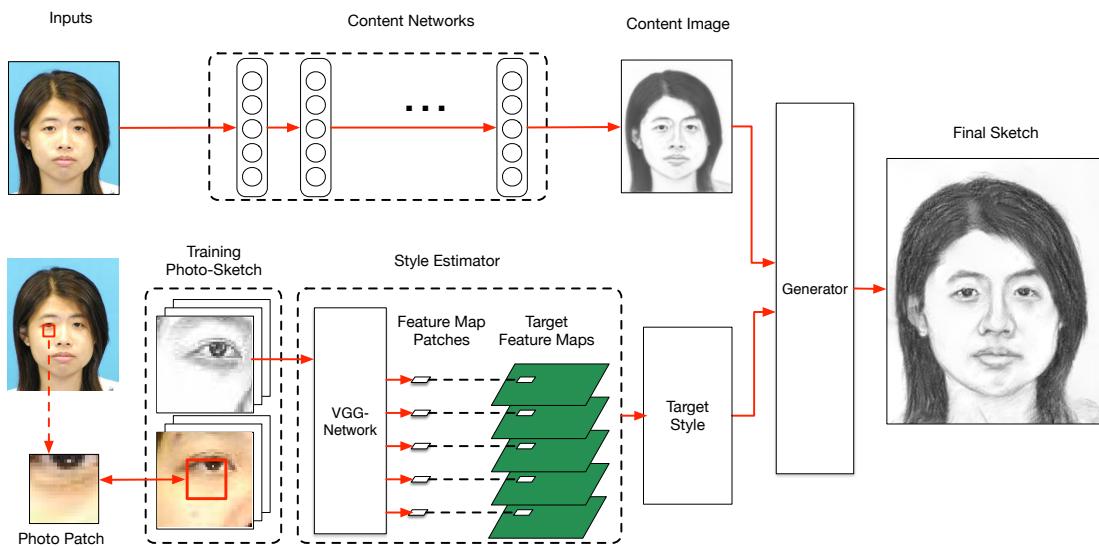
162 that the linear transformation holds better locally, and there-  
163 fore they proposed a patch based method to perform sketch  
164 synthesis. In [14], a MRF based method was proposed to  
165 preserve large scale structures across sketch patches. Vari-  
166 ants of the MRF based methods were introduced in [19, 20]  
167 to improve the robustness to lighting and pose, and to ren-  
168 der the ability of generating new sketch patches. In addition  
169 to these MRF based methods, approaches based on guided  
170 image filtering [11] and feed-forward convolutional neural  
171 network [18] are also found to be effective in transferring  
172 photos into sketches. A very recent work similar to ours is  
173 reported by Zhang *et al.* [17]. They proposed a two-branch  
174 FCNN to learn content and texture respectively, and then  
175 fused them through a face probability map. Although their  
176 results are impressive, their sketch textures do no look nat-  
177 ural and the facial components are over smoothed.

## 179 2.2. Style Transfer with CNN

180 Texture synthesis has long been a challenging task. Tra-  
181 ditional methods can only imitate repetitive patterns. Re-  
182 cently, Gatys *et al.* [4, 5] studied the use of CNN in style  
183 representation, and proposed a method for transferring the  
184 style of one image (referred to as the style image) to an-  
185 other (referred to as the content image). In their method,  
186 a target style is first computed based on features extracted  
187 from the style image using the VGG-Network. An output  
188 image is then generated by iteratively updating the content  
189 image and minimizing the difference between its style and  
190 the target style. Justin *et al.* [7] further accelerated this pro-  
191 cess by learning a feed forward CNN in the training stage.  
192 These methods represent styles by a multi-scale Gram ma-  
193 trix of the feature maps. Since the Gram matrix only cares  
194 about global statistics, local structures may be destroyed  
195 when the style image is very different from the content im-  
196 age. Although this may not be a problem in transferring  
197 artistic styles to images, this will definitely produce notice-  
198 able artifacts in the face sketch as people are very sensi-  
199 tive to the distortions of the facial features. In [3], Chen  
200 and Schmidt proposed a different patch based style transfer  
201 method which is better at capturing local structures. How-  
202 ever, it is still far from satisfactory to be employed in face  
203 sketch synthesis. Our style transfer approach is inspired by  
204 but different from the above work [4, 5, 7] in that our tar-  
205 get style is computed from image patches of many different  
206 images rather than from just one single image.

## 208 3. Style Representation

209 Following the work of [5], we use Gram matrices of  
210 VGG-19 [10] feature maps as our style representation. De-  
211 note the vectorized  $c$ th channel of the feature map in the  $l$ th  
212 layer of the final sketch  $\mathcal{X}$  by  $F_c^l(\mathcal{X})$ . A Gram matrix of  
213 the feature map in the  $l$ th layer is then defined by the inner  
214 product of all vectors in  $F_c^l(\mathcal{X})$ :

216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233

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

237  
 238 products between two channels of this feature map, i.e.,  
 239

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

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

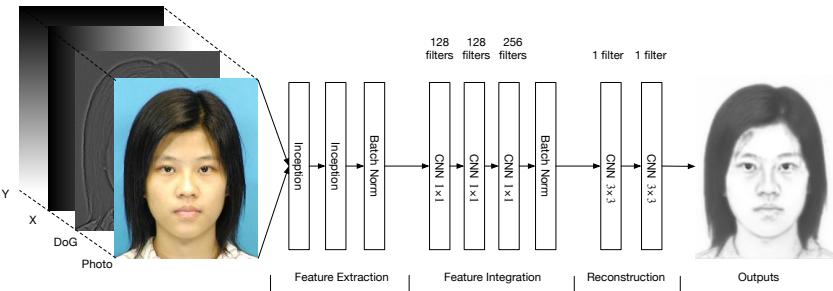
## 258 4. Methodology

259 Our method can be classified as a shading synthesis  
 260 method. The steps of our method are summarized in Fig. 2.  
 261 First, a preprocessing step as described in [14] is carried  
 262 out to align all photos and sketches in the training set by  
 263 the centers of the two eyes. An eye-aligned test photo  $\mathcal{I}$   
 264 is then fed into two branches, namely the content network  
 265 and the style estimator. The content network converts  $\mathcal{I}$

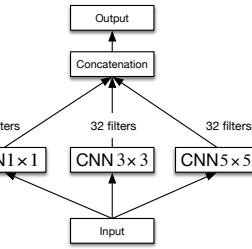
266 into a content image  $\mathcal{C}$ , which outlines the shape of the  
 267 face and the key facial features such as nose, eyes, mouth  
 268 and hair. The style estimator divides  $\mathcal{I}$  into a grid of non-  
 269 overlapping  $16 \times 16$  patches. For each test patch, it locates  
 270 the most similar photo patch from the photo-sketch pairs in  
 271 the training set and produce a target sketch patch from the  
 272 corresponding sketch in the pair. A pyramid column feature  
 273 (Section 4.2) is then computed for the target sketch patch.  
 274 Finally, a target style can be computed from a grid of these  
 275 pyramid column features, and a final sketch  $\mathcal{X}$  can be syn-  
 276thesized by applying the target style to  $\mathcal{C}$  through neural  
 277 style transfer [5].

### 278 4.1. Content Image Generation

279 The architecture of our content network is shown in  
 280 Fig. 3. Besides the test photo, we feed three extra chan-  
 281 nels containing the spatial information (i.e.,  $x$  and  $y$  coor-  
 282 dinates) and a difference of Gaussian (DoG) image into our  
 283 content network. As pointed out in [14], face sketch syn-  
 284 thesis algorithms can benefit from integrating features from  
 285 multiple resolutions. Hence, we employ an inception mod-  
 286ule [12] for feature extraction, which concatenates features  
 287 extracted using a two-layer-inception module are then fed into  
 288 a three-layer-CNN for feature integration, where all filters  
 289 have a size of  $1 \times 1$ . Finally, the integrated features are used  
 290 to reconstruct the content image  $\mathcal{C}$  by a two-layer-CNN with  
 291 the filter size being  $3 \times 3$ . Since  $L_1$ -norm is better at pre-  
 292 serving details than  $L_2$ -norm, we use the  $L_1$ -norm between  
 293  $\mathcal{C}$  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.,

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

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

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

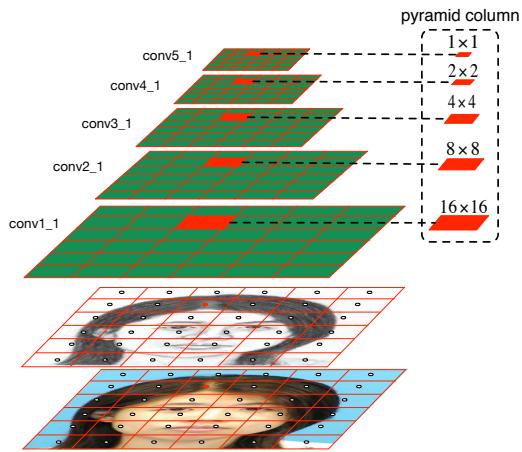


Figure 4. Illustration of the *pyramid column feature*. After finding a target sketch patch, we can extract its corresponding feature patches from the original feature maps. These five feature patches make up the *pyramid column feature*.

where  $\alpha$ ,  $\beta$  and  $\beta_2$  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 :

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

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

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

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

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

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

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

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 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  $288 \times 288$  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 shortcomings of existing quantitative evaluation criteria, we also propose the Normalized Gram Matrix Difference (NGMD) as a new effective metric for quantitatively evaluating the performance of face sketch synthesis.

### 5.1. Direct Style Transfer

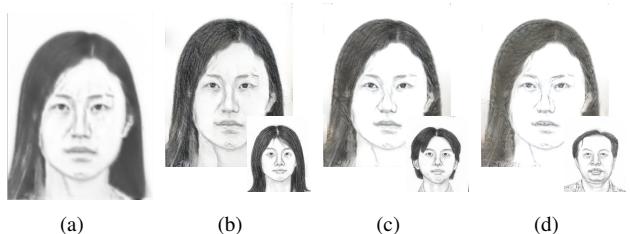


Figure 5. (a) is the content image generated by our content network. (b), (c) and (d) are generated by different styles. It can be seen that the generated sketch becomes brighter as the amount of hair decreases in the style image.

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. 5. 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. 5(c) and 5(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. 5(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.

### 5.2. Qualitative Evaluation

Fig. 6 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,

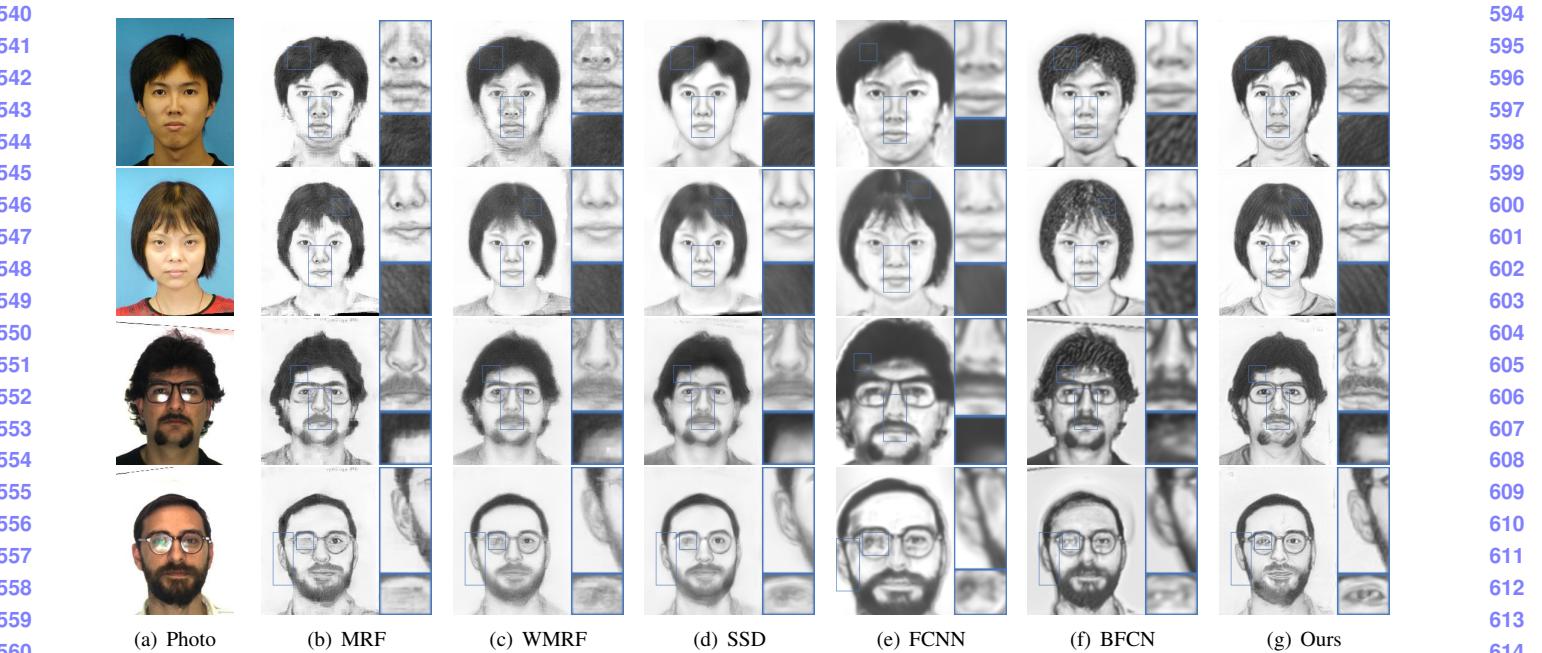


Figure 6. Qualitative evaluation of the face sketch synthesis results on the CUHK (first two rows) and AR (last two rows) datasets. (a) Original Photos. (b) MRF [14] (c) WMRF [20] (d) SSD [11] (e) FCNN [18] (f) BFCN [17] (g) Ours. The proposed method preserves more texture details (e.g., in the hair and nose), and is also best at keeping the origin structures (e.g., glasses) in the photos.

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. 6). In contrast, our method can well maintain the image content and create sharp edges.

### 5.3. Quantitative Evaluation

**Sketch Recognition** Sketch synthesis methods are commonly 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

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

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%

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.

**Normalized Gram Matrix Difference (NGMD)** However, the evaluation based on face sketch recognition is not effective since every method achieves a very high (close to 100%) recognition rate. Zhang *et al.* proposed a Multiscale Pixel-wise Reconstruction Loss (MPRL) in [18]. Although we agree that the generated sketches will be identical to the ground truth when MPRL is zero, there are two reasons why it is not a good criteria. First, experimental results of [18, 17] show that  $L_2$ -norm would give blurry sketches. Second, the ground truth drawn by human usually does not

648 Table 2. Averaged NGMD value of different methods at different level on AR and CUHK datasets. A smaller NGMD indicates the 702  
 649 generated texture is more similar to the ground truth. 703  
 650 704

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	<b>0.0035</b>	<b>0.008</b>	<b>0.029</b>	<b>0.08</b>	<b>0.17</b>	<b>0.007</b>	<b>0.012</b>	<b>0.033</b>	<b>0.07</b>	<b>0.12</b>

660 correspond exactly with photos. Hence, it is extremely difficult, if not impossible, for algorithms to produce result close 712  
 661 to the ground truth. 713

662 Meanwhile, human can often be able to tell whether a 714  
 663 face sketch is generated by an algorithm or drawn by an 715  
 664 artist from just a quick glance without examining it in detail. 716  
 665 This indicates that there exists a big gap between the style 717  
 666 of algorithm-generated sketches and that of those drawn by 718  
 667 artists. To quantitatively evaluate the style similarity 719  
 668 between a generated sketch and the sketch drawn by a real 720  
 669 artist, we introduce the normalized Gram matrix difference 721  
 670 (NGMD) between the generated sketch  $\mathcal{X}$  and the drawn 722  
 671 sketch  $\tilde{\mathcal{X}}$ : 723

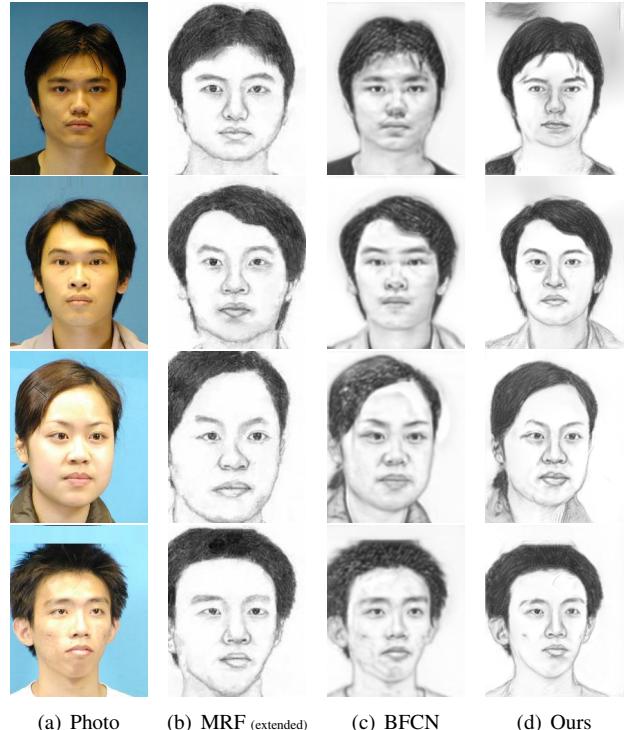
$$D_s^l \left( \mathcal{X}, \tilde{\mathcal{X}} \right) = \frac{\left\| G^l \left( \mathcal{X} \right) - G^l \left( \tilde{\mathcal{X}} \right) \right\|_2^2}{\left\| G^l \left( \tilde{\mathcal{X}} \right) \right\|_2^2} \quad (8)$$

678 where  $l \in L_s$ . The smaller the NGMD value is, the more 724  
 679 similar in style  $\mathcal{X}$  and  $\tilde{\mathcal{X}}$  are. 725

680 Table 2 shows the average NGMD values for different 726  
 681 methods. [18] can only roughly outline the face but 727  
 682 cannot add textures, and therefore it has a high average 728  
 683 NGMD value accordingly. The results of MRF [14] and 729  
 684 WMRF [20] are more similar to hand drawn sketches in 730  
 685 style as they use patches of hand drawn sketches to compose 731  
 686 their final sketches. They therefore have a smaller average 732  
 687 NGMD value as expected. Inheriting the ability of denoising, 733  
 688 the filtering based approach [11] is good at suppressing 734  
 689 noises in the results, but at the same time over-smooths the 735  
 690 results. It therefore has an average NGMD value higher 736  
 691 than that of WMRF. 737

## 693 5.4. Generalization

695 **Light and Pose Invariance** As discussed in [19], light 740  
 696 and pose changes may influence the result a lot. We choose 741  
 697 several photos from [19] and compare our results with 742  
 698 MRF (extended) [19] and BFCN [17]. As seen from the 743  
 699 comparison in Fig. 7, our proposed method is not influenced 744  
 700 by pose and light changes, and can generate good textures 745  
 701 under a laboratory environment. 746



738 Figure 7. Experiment under different light and pose settings. 739

**Real World Photos** We further test the robustness of our 740  
 model on some real world photos, and the results are shown 741  
 in Fig. 8. The first two rows are Chinese celebrity faces 742  
 from [19], and the latter two come from the world wide 743  
 web. Since the test photos may not be well aligned, we just 744  
 turn off the component loss. The parameters we use here 745  
 are  $\alpha = 0.004$ ,  $\beta_1 = 1$ ,  $\beta_2 = 0$ . Although the background 746  
 is cluttered and the positions of the faces are not strictly 747  
 constrained, the hair styles of our results are still clear and 748  
 sharp, whereas FCNN and BFCN fail to produce good tex- 749  
 tures. 750

## 752 5.5. Effectiveness of the model

753 The loss function we minimize during the generation 754  
 of sketches contains three terms for content, style and key 755

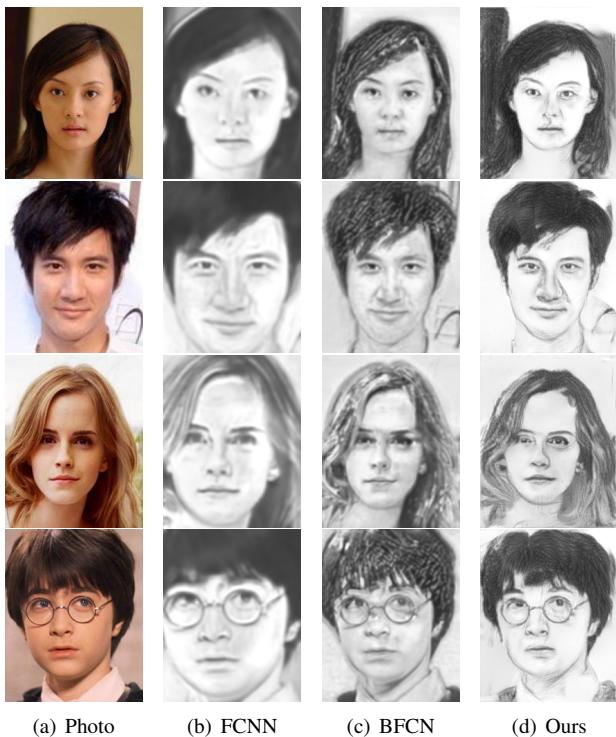
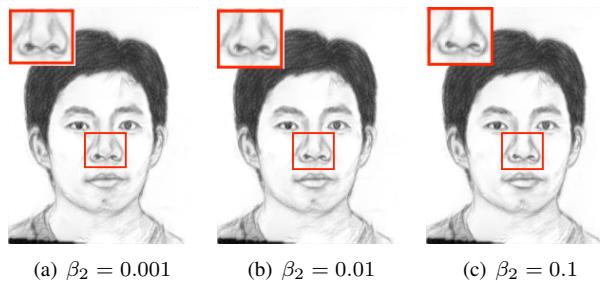


Figure 8. Experiments with real world photos.

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.

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. 10 shows that the sketch with style transferred contains more textures and looks 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 the feature maps and calculate the gradient of Eq. (3), which is computationally intensive.

## 6. 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 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](#)
- [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* [860](#)



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

Figure 10. Final results for sketch synthesis with fixed  $\alpha = 0.004$  and different  $\beta_1$  and  $\beta_2$  values.

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

918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971