

Face Sketch Synthesis by Style Transfer with Local Features

Anonymous ICCV submission

Paper ID 621

Abstract

Face sketch synthesis is challenging as it is difficult to generate sharp and detailed textures. In this paper, we propose a new framework based on deep neural networks. Imitating the process of how artists draw sketches, our framework synthesizes face sketches in a cascaded manner in which a content image is first generated that outlines the shape of the face and key facial features, and textures and shadings are then added. We utilize a Fully Convolutional Neural Network (FCNN) to create the content image, and propose a local feature based style transfer to append textures. The local feature, what we call pyramid column feature, is a set of features at different convolutional layers corresponding to the same local sketch image patch. We demonstrate that our pyramid column feature can not only preserve more sketch details than common style transfer method but also surpass traditional patch based approach. Our model is trained on [??](https://deepart.io/) training data set and evaluated on other datasets. Quantitative and qualitative evaluations suggest that our framework outperforms other state-of-the-arts methods. In addition, despite of the small training data ([??](https://deepart.io/) face-sketch pairs), our model shows great generalization ability across different datasets and can generate reasonable results under practical situations.

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 sketch has 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 ex-

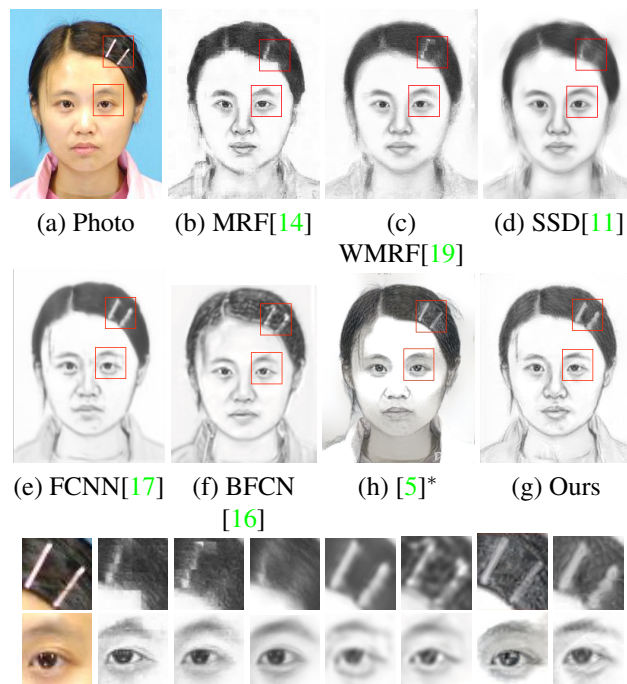


Figure 1. Face sketches generated by existing methods and the proposed method. Our method can not only preserve both hair and facial content, but also maintain sharp textures. (h)* is obtained from deep art website¹ by using the photo as content and a sketch from training set as style.

emplar based methods [14, 11, 18, 19] were proposed. In these methods, an input photo is divided into patches and candidate sketches for each photo patch are selected from a training set. The main drawback of such kind of methods is that if the test image can't find a similar patch in the training set, they may lose some contents in the final result. For example, the sketches in the first row of Fig. 1 fail to keep the hairpins. Besides, some methods [11, 19] clear away the textures when they try to eliminate the inconsistency between neighboring patches. Another potential risk is that the result may not look like the original photo, e.g. left eye in Fig. 1 (b). Recently, approaches [16, 17] based on convolutional neural network (CNN) were developed to solve these problems. Since these models directly gener-

¹<https://deepart.io/>

ates sketches from photo, they can maintain the structures and contents of the photos. However, the loss function of them are usually mean square error (MSE) or variation of it, which is responsible for the blur effect, *e.g.* Fig. 1 (e) and (f). The reason is that MSE prefers values close to mean, and is not suitable for texture representations. The popular neural style transfer provides a better solution for texture synthesis. But there are two obstacles towards directly applying such kind of method. First, it is easily influenced by illumination of the photo, see the face of Fig. 1 (h). Second, it needs a style image to give the global statistics of textures. If the given style doesn't coincide with target sketch (which we don't have), some side effects will occur, *e.g.* the nose in Fig. 1 (h). Extensive experiment and discussion is given in Section ??.

For an artist, the procedure of sketching a face usually starts with outlining the shape of the key facial features like the nose, eyes and mouth. Textures and shadings are then added to regions such as hair lips, and bridge of the nose to give sketches a specific style. Inspired by this and neural style transfer [4], we propose a new framework for face sketch synthesis that can overcome the aforementioned limitations. In our method, the outline of a face is delineated by a feed-forward neural network, and textures and shadings are then added by a style transfer approach. Specifically, we design a new architecture of Fully Convolutional Neural Network (FCNN) which contains inception layers [12] and convolution layers with batch normalization [6] to outline the face (Section ??). For the texture part, we first divide the feature maps of the target sketch in each layer into a fixed size grid and combine features from different layers but at the same grid location into a pyramid feature column (Section ??). These pyramid feature columns can be generated by local sketch patches from the training set. A target style is then computed by assembling these pyramid columns. These sketch patches are found by matching the test **content** patch to the **content-sketch** pairs in training set (Section ??). 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 well preserve the content of the test photo.

2. Related Work

2.1. Face Sketch Synthesis

Based on the taxonomy of previous studies [11, 19], face sketch synthesis methods can be roughly categorized into profile sketch synthesis methods [1, 2, 15] and shading sketch synthesis methods [9, 11, 13, 14, 17, 18, 19]. 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 face sketches from face photos. This assumption, however, does not always hold since the modality of face photos and that of face sketches are quite different. Fortunately, Liu et al. [9] found that the linear transformation holds better locally and therefore they proposed a patch based method to perform sketch synthesis. A MRF based method [14] was proposed to preserve large scale structures across sketch patches. Variants of the MRF based methods were introduced in [18, 19] 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 [17] are also found to be effective in transferring photos into sketches. A very recent work similar to ours is done by Zhang *et al.* [16]. They proposed a two branch FCNN to learn content and texture respectively and then fusion them through a face probability map. Although their results are impressing, the sketch texture is not natural and the facial components are smoothed.

2.2. Style Transfer with CNN

Texture synthesis has long been a challenging task. Traditional method can only imitate repetitive patterns which has a strong limitation. Recently, Gatys *et al.* [4, 5], studied the use of CNN in style representation (including texture and color) where a target style is computed based on features extracted from an image using the VGG-Network and an output image is generated by minimizing the difference between its style and the target style. It can transfer any style to any images, and the results are impressive. Justin *et al.* [7] further accelerated this process by learning a feed forward CNN in the training stage. These methods represent textures by a multi-scale gram matrix of feature maps. Since gram matrix cares more about global statistics, if the style is very different from the photo, it usually breaks the local structures. Although it is not a big deal in artistic style, it can't be tolerated in face sketch synthesis. In [3], Chen and Schmidt propose a different patch based style transfer method which is better at capture local structures. However, it is still not suitable for this task. Our style transfer mechanism is inspired by but different from these works [4, 5, 7] in that our target style is extracted from many image patches rather than from a single style image. Note that there usually does not exist a single style image in the training set that matches all properties of the test image.

3. Motivation of Pyramid Feature Column

Following the practice of [5], we use the gram matrix of VGG-19[10] feature maps as our style representation. Denote the vectorized feature map of the final sketch \mathcal{X} in the

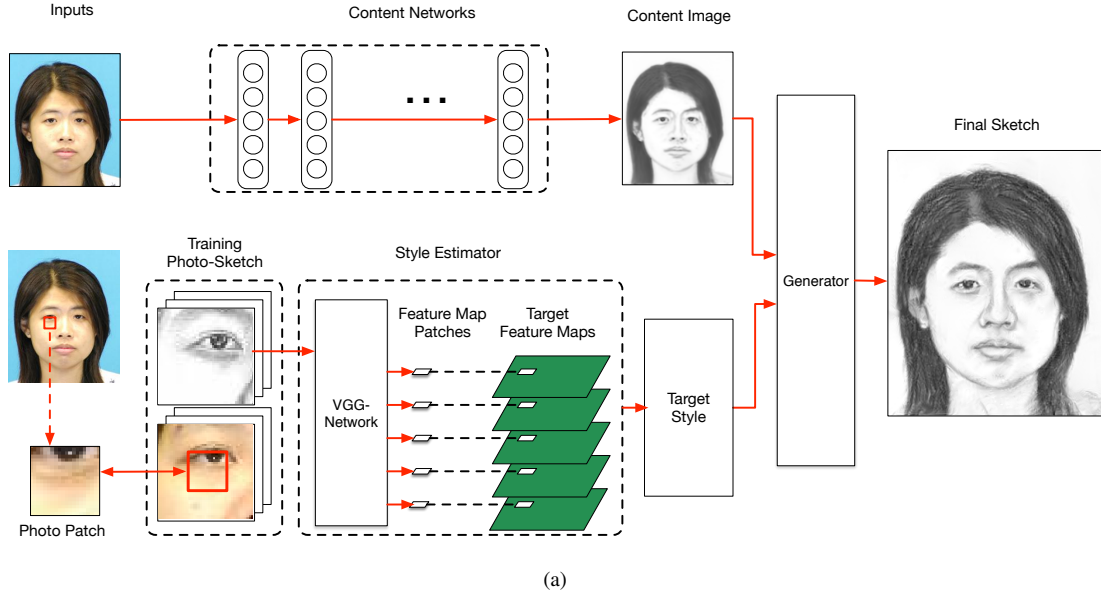


Figure 2. The proposed method contains two branches which take an photo aligned by the eyes as inputs. The content network outputs a content image and the style estimator generates a target style. The final sketch is generated by combining the target style with the content image. *The figure may need to revise, in order to show the optimization process of target sketch.*

l th layer by $F^l(\mathcal{X})$. A gram matrix is the inner product between the feature maps in l th layer

$$G_{ij}^l(\mathcal{X}) = \sum_{k=1}^{M_l} F_{ik}^l(\mathcal{X}) F_{jk}^l(\mathcal{X}) \quad (1)$$

where $G^l(\mathcal{X}) \in \mathcal{R}^{N_l \times N_l}$, M_l is the height times width of the feature map $F^l(\mathcal{X})$, and N_l is the number of feature maps in the l th layer. Since $G_{ij}^l(\mathcal{X})$ is an inner product of feature maps, a gram matrix is actually a summary statistics of feature maps discarding the spatial information. Although we still not clear what these values exactly mean, we can safely make an assumption that it at least captures the density distribution of a sketch. In other word, if the given sketch style has much less hair than the test image, the generated sketch \mathcal{X} will possibly be unnaturally bright than a natural sketch. Experiments in Section ?? also prove this. Thus it is important to keep the given style sketch roughly the same with test image statistically. On the other hand, there usually does not exist a candidate image in the training set that perfectly matches a given test photo in style. We hence propose a feature level patch based method to estimate the style of the final sketch. Each feature patch corresponds to a sketch patch. The reason why we can separate features into patches comes from [8]. The feature vectors at different position of feature map can be viewed as independent samples when we use gram matrix.

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 for all photos and sketches in a training set to align the centers of two eyes. A test photo \mathcal{I} is then fed into two branches, namely the content network and the style generator. The content network converts the test photo into a content image \mathcal{C} , where the shape of the face are outlined with the key facial features preserved, such as noses, eyes, mouths and hair. The style estimator takes a 16×16 local patch from test photo as input and searches photo-sketch pairs (*Maybe we can find a content-sketch pairs instead, because real photo varies too much*) in the training set to find a target sketch patch S_{ij} ((i, j) denotes the patch location). Each S_{ij} with its surrounding region can generate a pyramid feature column U_{ij} . Combining all U_{ij} , we can get the target style features of \mathcal{I} , i.e. \tilde{U} . Given \mathcal{C} and \tilde{U} , we can generate a sketch \mathcal{X} that combines the content information in \mathcal{C} with the style representation \tilde{U} following the iterative procedure in [5].

4.1. Content Image Generation

Our content network architecture is shown in Fig. 3. In addition to the test photo, we feed two extra channels containing spatial information (i.e., x and y coordinates) and a difference of Gaussian (DoG) image into the content network. As pointed out in [14], face sketch synthesis algorithms benefit from integrating features from multiple res-

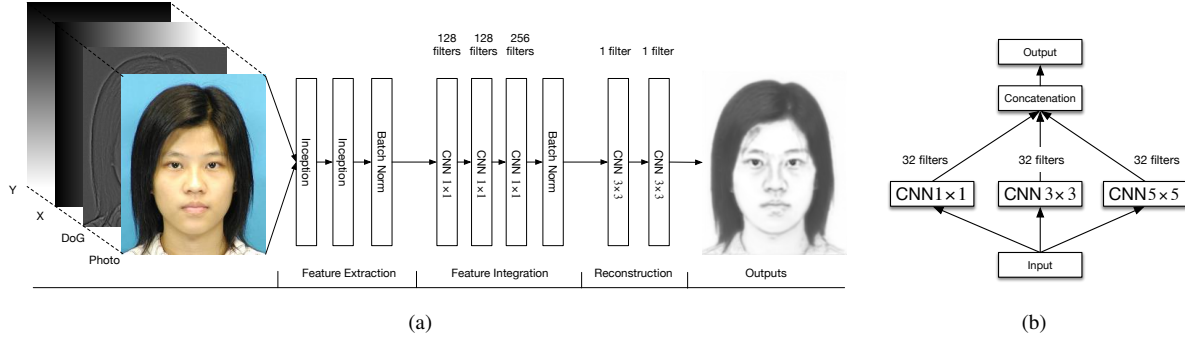


Figure 3. Illustration of the content network for generating a content image. The numbers above the building block denote the number of CNN filters. (a) The architecture of content network. (b) The inception module in (a) contains three groups of filters with different sizes.

olutions. We employ an inception module inspired by the GoogLeNet [12] to extract features. It concatenates feature maps generated from filters with different spatial resolutions. Our inception unit contains 3 different size of filters (1×1), (3×3) and (5×5) (see Fig. 3(b)). Then, the output features are fed to a three-layer-CNN for feature integration, where the size of all filters are fixed at 1×1 . Finally, the integrated features are used to reconstruct the content map by a two-layer-CNN with the filter size being 3×3 . A mirror padding is carried out before the convolution operation when necessary to ensure the output feature map is the same size as the input. The output content image \mathcal{C} is a gray image with size 250×256 .

4.2. Pyramid Feature Column

Fig. 4 shows an example of the pyramid feature column. Denote the feature maps (of the l th layer) used to estimate the style of the final sketch by A^l . In our feature patch based method, we divide A^l into a fixed size of grid. Due to the different feature map size, the sizes of the feature patches at layer $conv1_1$, $conv2_1$, $conv3_1$, $conv4_1$ and $conv5_1$ are 16×16 , 8×8 , 4×4 , 2×2 and 1×1 . The photos and sketches are resized to 288×288 thus the size of grid is 18×18 . Grouping feature map patches having the same grid indexes (i, j) at different layers together, we get a pyramid feature column U_{ij} . To estimate U_{ij} , a sketch patch in the training set is fed to the VGG-Network and a pyramid column is composed of the resulting feature maps. This process consists of two steps: (1) find a matching sketch patch S_{ij} from the training set for U_{ij} and (2) feed S_{ij} to VGG-Network and extract U_{ij} from the resulting feature maps.

Find Matching Sketch Patch Unlike previous works [14, 19], we examine the similarity of content patches to find a matching sketch S_{ij} for U_{ij} . Compared with photo patches, the content patches are more easier to match. Given a test photo patch \mathcal{I}_{ij} ,

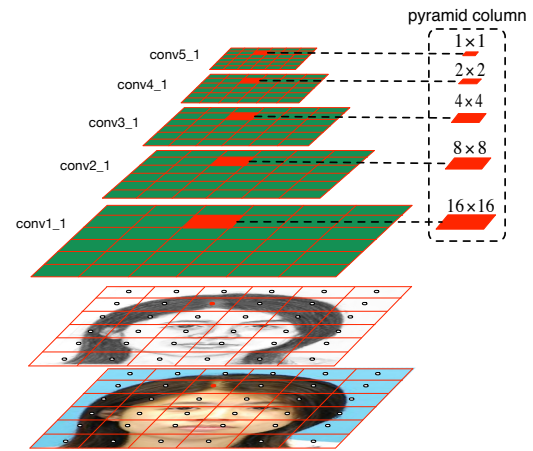


Figure 4. Illustration of pyramid feature column. Feature maps of the final sketch A^l are divided into a fixed 18×18 grid. A pyramid column U_{ij} consists of feature map patches at different layer having the same grid indexes (i, j) .

Estimate Pyramid Column We compose an estimate for U_{ij} from the resulting feature maps obtained by feeding S_{ij} to the VGG-Network. Specifically, after feeding S_{ij} to the VGG-Network, we form a hyper-grid \tilde{U} on the resulting feature maps. The pyramid column corresponding to \tilde{C}_{ij} is selected as an estimation to U_{ij} . This pyramid column is the cell that contains \tilde{C}_{ij} when projecting S_{ij} onto \tilde{U} .

After A^l are estimated, we calculate the target gram matrices by: $\{G_t^l = A^l \cdot (A^l)^T\}$ where $l \in L_s$.

4.3. Loss Function

To generate a sketch that combines the content image with the estimated style, we adopt the methodology described in [4]. Specifically, we minimize a loss function consisting of a style loss, a content loss and a component loss. The style loss is defined as the difference between the gram matrix of the final sketch and the target gram matrix

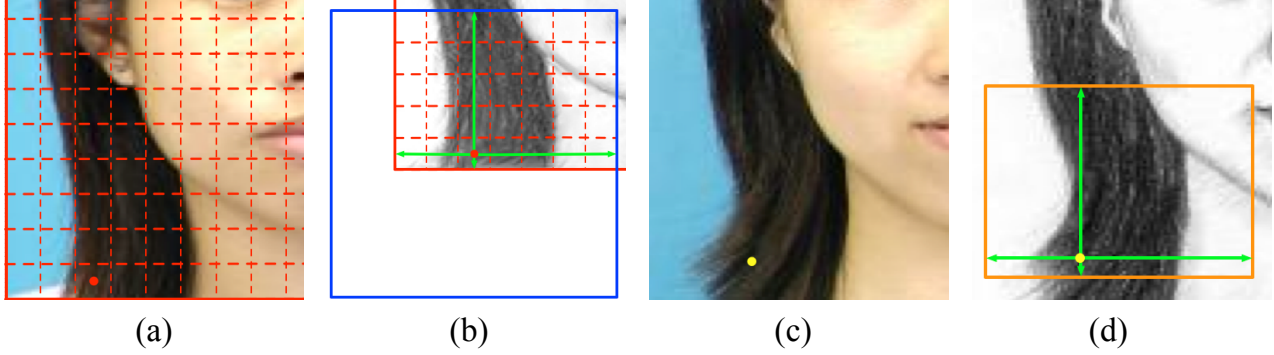


Figure 5. Find a matching patch for boarder cells. (a) The center C_{ij} of a boarder cell is denoted as the red dot in the photo. (b) The region associated with U_{ij} is the intersection of the blue square with the sketch. The distances between C_{ij} to the intersection borders are denoted by green lines. (c) \tilde{C}_{ij} denoted as a yellow dot is located by examining photo similarity. (d) The distances of \tilde{C}_{ij} to borders of S_{ij} equal to the distances of C_{ij} to borders of intersection region in (b) such that the location of \tilde{C}_{ij} in S_{ij} is the same as the location of C_{ij} in the region associated with U_{ij} .

i.e.,

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

where N_l denotes the number pixels in the feature map at layer l . The content loss is defined based on the difference between the feature map of the sketch and that of the content image at layer conv1_1:

$$\mathcal{L}_c(\mathcal{X}) = \|F^{\text{conv1.1}}(\mathcal{X}) - F^{\text{conv1.1}}(\mathcal{C})\|_2^2. \quad (3)$$

Human is able to distinguish different people from key components such as eyes, nose and mouth, which indicates that these features are the most discriminative parts of a face. Specific styles or textures are usually used to emphasize these components, for example, sharp edges with shadings at the two sides of the nose are used to convey 3D information. To better transfer styles of these components, we employ a component loss to encourage the key component style of the final sketch being the same as the target key component style. Since two eyes are placed at fixed positions, the key components lie roughly within a rectangular region taking the positions of two eyes as vertices. Key component style is given by gram matrices calculated within feature map regions \mathcal{R} corresponding to the key components. These regions are specified by hyper-grid cells U_{ij} whose C_{ij} are inside the rectangular region. More specifically, a target style for the key component is calculated: $\{\hat{G}_t^l = A_c^l \cdot (A_c^l)^T\}$ where A_c^l is the composed feature map patch in A^l corresponding to the key components, $l \in L_s$. The style of the key components in final sketch $\hat{G}^l(\mathcal{X})$ is calculated in exactly the same manner. The component loss is hence defined as:

$$\mathcal{L}_k(\mathcal{X}) = \sum_{l \in L_s} \frac{1}{M_l^2 \hat{N}_l^2} \|\hat{G}^l(\mathcal{X}) - \hat{G}_t^l\|_2^2 \quad (4)$$

where \hat{N}_l denotes the number of pixels in the feature map region \mathcal{R} at layer l . The total loss we minimize is

$$\mathcal{L}_t(\mathcal{X}) = \alpha \mathcal{L}_c + \beta_1 \mathcal{L}_s + \beta_2 \mathcal{L}_k \quad (5)$$

where α , β_1 and β_2 are the weighting factors for content, style and component losses respectively. 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.

4.4. Implementation Details

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

$$W^k = W_r^k + W_g^k + W_b^k \quad (6)$$

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.

5. Final copy

You must include your signed IEEE copyright release form when you submit your finished paper. We MUST have this form before your paper can be published in the proceedings.

Please direct any questions to the production editor in charge of these proceedings at the IEEE Computer Society Press: Phone (714) 821-8380, or Fax (714) 761-1784.

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, 4
- [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
- [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] Y. Li, N. Wang, J. Liu, and X. Hou. Demystifying neural style transfer. *CoRR*, abs/1701.01036, 2017. 3
- [9] 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
- [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
- [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, 4
- [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
- [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] 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
- [17] 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
- [18] 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
- [19] 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