Face ~~Sketch Synthesis by Style Transfer~~ Sketching with ~~Local Features~~Deep Neural Networks

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 face sketch synthesis 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 training data set and evaluated on other*~~ *By exploiting the capability of deep networks for generating beautiful textures and stylized images, our framework can produce sketches that look like sketches drawn by real artists in strokes. Experiments are carried out on both the CUHK and AR datasets. Quantitative and qualitative evaluations suggest that our framework outperforms other state-of-the-arts methods.* ~~*In addition, despite of the small training data (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.

~~(a) Photo (b) MRF[15] (c) WMRF[20] (d) SSD[12] (e) FCNN[18] (f) BFCN [17] (h) [5]* (g) Ours 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 website by using the photo as content and a sketch from training set as style.~~

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 ~~[15, 12, 19, 20]~~ [15, 19, 20] 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.~~**??** ~~fail to keep the hairpins. Besides, some methods [12, 20] clear away the textures when they try to eliminate the~~ sketch output of a photo patch can be represented by either the best candidate or a linear combination of the best K candidate sketch patches. However, processing each photo patch individually often introduces inconsistency between neighboring patches. ~~Another potential risk is that the result may not look like the original photo, left eye inFig.~~ **??** ~~(b)~~ To tackle this problem, methods in [15, 19, 20] adopted Markov Random Fields (MRF) to regularize the synthesis process. Recently, ~~approaches [17, 18] based on~~ an image based approach [12] based on the idea of joint image filtering was proposed. It was found superior in suppressing noises in the sketch. Another recent approach based on a feed-forward convolutional neural network (~~CNN) were developed to solve these problems. Since these models directly generates 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)~~

0

or variation of it, which is responsible for the blur effect, Fig. ?? (e) and (f). The FCNN) was found to be more effective in generating discriminative facial details [18]. Although sketches produced by these methods do look like the faces in the photos, they look quit different from sketches drawn by real artists. One major 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.?? (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, the nose these methods do not consider the style of the synthesized sketches. For example, the hair in their synthesized sketches is often lack of texture, and the region around the face boarders often suffers from blurring artifacts (see Fig. ??). Besides, both the exemplar based and image based approaches cannot handle structures that do not exist in the training set (e.g., clothes in Fig.?? (h).Extensive experiment and discussion is given in Section .

??).

Inspired by the way how artists draw sketches, we propose a new framework for face sketch synthesis that can overcome the aforementioned limitations. For an artist, instead of drawing sketches region by region, 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 [13] and convolution layers with batch normalization [6] to outline the face (Section). For the texture part, we first We formulate the style as a statistical measure of the features extracted from multiple resolutions of the sketches using the VGG-Network [11]. Due to the non-linearity of these features and the fact that there usually does not exist a training photo that is very similar to the test photo, estimating the style for a target sketch is not easy. In spired by the work of Wang and Tang [15], we 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 setinto a grid and use the features of a candidate sketch patch as a surrogate for the feature map patch of a grid cell. A target style is then computed by assembling these pyramid columns. These sketch patches are found by matching the test patch to the pairs in training set(Section)from the resulting feature maps. Our approach is superior to the current state-of-the-art methods in that It (1) it is capable of generating more stylistic sketches without introducing over smoothing artifacts It can well preserve the content of the test photo.

and (2) it can draw structures which do not exist in the training set.

2. Related Work

2.1. Face Sketch Synthesis

Based on the taxonomy of previous studies [12, 20], face sketch synthesis methods can be roughly categorized into profile sketch synthesis methods [1, 2, 16] and shading sketch synthesis methods [9, 12, 14, 15, 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 [14] 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 [15] 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 [12] 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 done by Zhang [17]. 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

2.2. Style Transfer with Convolution Neural Network

The class of Convolutional Neural Networks (CNN) is perhaps the most powerful tool in image processing. It

usually contains layers of filters each of which extracts a certain feature from the input or from the output of the previous layer. The VGG-Network [11], one popular instance of such networks, rivals human performance in image classification tasks. This demonstrates the ability of CNN in feature extraction. In [4, 5], Gatys et al. 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 [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 inartistic 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.~~ Likewise, a perceptual loss function measuring the difference in style between a targeting image and images generated from a CNN was proposed in [7] and it was then exploited in the CNN training stage. 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~~ images 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. Hence, we propose computing the target style based on multiple images. The difficulty of generating a target style from multiple images lies in the non-linearity of the neural network.

~~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 combing the target style with the content image.~~

3. ~~Motivation of Pyramid Feature Column~~

~~Following the practice of [5], we use the gram matrix of VGG-19[11] feature maps as our style representation. Denote the vectorized feature map of the final sketch $\mathcal{X}$ in the lth layer by $F^l(\mathcal{X})$. A gram matrix is the inner product between the feature maps in lth layer~~

$$G^l_{ij}(\mathcal{X}) = \sum_{k=1}^{M_l} F^l_{ik}(\mathcal{X}) F^l_{jk}(\mathcal{X})$$

~~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 lth layer. Since $G^l_{ij}(\mathcal{X})$ 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.~~

3. ~~Methodology~~Overview of Our Method

Our method can be classified as a shading synthesis method. The steps of our method are summarized in Fig. **??**. First, a preprocessing step as described in [15] 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, respectively. The content network converts the test photo into a content image $\mathcal{C}$, where facial features and the shape of the face are outlined with the ~~key~~ global arrangement of the facial features preserved~~,~~. It is not necessary that the style of the content image resembles a real sketch, but the shapes of different features such as noses, eyes, mouthes and hair, should be delineated. The style estimator takes ~~a 16×16 local patch from~~ the test photo as input and ~~searches~~ utilizes photo-sketch pairs ~~()~~ 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 $\mathcal{I}$, $\tilde{U}$~~generate a target style $\mathcal{S}$ which we want to transfer to the final sketch. Given $\mathcal{C}$ and ~~$\tilde{U}$, we can~~ $\mathcal{S}$, we 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].~~

3.1. ~~Content Image Generation~~

~~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.~~

~~Our content network architecture is shown in Fig. ??~~$\mathcal{S}$. This is formulated as an optimization problem over an energy function taking both $\mathcal{C}$ and $\mathcal{S}$ into considerations. Details will be explained in the following subsections.

4. Content Image Generation

We use CNN to build our content network (see Fig **??**). In addition to the test photo, ~~we feed two extra~~ two location channels containing spatial information (i.e., x and y coordinates) and a difference of Gaussian (DoG) ~~image~~ channel containing edge information are fed into the content network. As pointed out in [15], face sketch synthesis algorithms benefit from integrating features from multiple resolutions. We employ an inception module inspired by the GoogLeNet [13] to extract features. It concatenates feature maps generated from filters with different spatial resolutions. Our inception unit contains 3 CNN filter groups, with each group having 32 filters. The filters in these 3 ~~different size of filters (1×1), (3×3) and (5×5)~~ groups have a size of $autoref(1\times 1)$, $autoref(3\times 3)$ and $autoref(5\times 5)$ respectively (see Fig. **??**3(b)). ~~Then, the~~ The output features are then fed to a three-layer-CNN for feature integration, where the size of all filters are fixed at ~~1×1~~$autoref(1\times 1)$. 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.~~ It is observed that the output of the content net outlines the shapes of the facial features, but its style does not look like sketches. For example, there exists no texture in the hair region and there is a lack of shadings to convey 3D information.

5. Sketch Style Generation

5.1. ~~Pyramid Feature Column~~Style Representation

The style features of sketches are extracted using the VGG-Network [11], a CNN that was originally designed for visual object recognition tasks, and was later used for texture feature representation in [4]. Here we use the intermediate feature maps extracted by the convolutional layers and max-pooling layers in the VGG-19 network just before the fully connected layers.

~~Fig. ?? shows an example of the pyramid feature column.~~ In the VGG-Network, convolutional layers extract features and max-pooling layers down-sample the features such that the following convolutional layers extract features in a lower resolution. We extract the outputs of the first convolutional layers at different resolutions for the computation of style. Following the terminology in [4], these layers are referred to as "conv1_1", "conv2_1", "conv3_1", "conv4_1", and "conv5_1", respectively, from the lowest level to the highest level. Define $L_s = \{\text{conv1\_1, conv2\_1, conv3\_1, conv4\_1, conv5\_1}\}$. 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

$$W^k = W_r^k + W_g^k + W_b^k \tag{1}$$

where W_r^k, W_g^k, and W_b^k are weights of the kth filter in the first convolutional layer for the R, G and B channels respectively, and W^k is the weight of the kth filter in the first convolutional layer of our modified network. Note that we keep the bias in our modified network the same as that in the original VGG-Network. Denote the feature map of the final sketch $\mathcal{X}$ in the lth layer (in matrix form) by $F^l autoref(\mathcal{X})$, where the element at the mth row and the nth column is the activation of the mth filter at location n. We use the gram matrix as our style representation. A gram matrix is a summary statistics of feature maps discarding the spatial information, and is originally designed for texture synthesis [4]. The gram matrix at layer l is given by:

$$G^l autoref(\mathcal{X}) = F^l autoref(\mathcal{X}) \cdot autoref(F^l autoref(\mathcal{X}))^T \tag{2}$$

where $G^l autoref(\mathcal{X}) \in \mathcal{R}^{M_l \times M_l}$, and M_l denotes the number of filters in layer l. A multi-resolution style representation is obtained by combining the gram matrices at different layers: $\mathcal{S} autoref(\mathcal{X}) = \{G^l autoref(\mathcal{X})\}$ where $l \in \{\text{conv1\_1, conv2\_1, conv3\_1, conv4\_1, conv5\_1}\}$. This style information is useful in guiding the sketch synthesis, especially in the texture region such as hair and shadings around the nose, eyes and mouth.

5.2. Target Style Generation via Multi-resolution Grouping

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 patch based method to estimate the style of the final sketch. Denote the feature maps (of the lth 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~~grid where each grid cell (feature map patch) at layer $conv5\_1$ contains only one pixel. Since the pooling size is 2×2, we choose the sizes of feature map patches at the next lower layer twice of that at the current layer. Thus, the sizes of the feature map 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~~, respectively. We introduce a multi-resolution feature descriptor by grouping feature

map patches having the same grid ~~indexes (i, j) at different layers together , we get a pyramid feature column~~indices i, j at different layer index l together as a pyramid column, U_{ij}. The photos and sketches are resized so that both the number of rows and columns are dividable by 16. A^l can then be represented as a hyper-grid U with the cells being U_{ij}. Similarly, we divide the test photo and the final sketch, respectively, into a grid with the same dimension as U. Denote the centers of these patches as C_{ij}. U_{ij} is hence a multi-resolution feature descriptor for a sketch region centered at C_{ij} (see Fig. **??**). To estimate U_{ij}, a sketch patch in the training set is fed to the VGG-Network and a pyramid column is composed ~~of~~ from the resulting feature maps. This process consists of two steps: (1) find a matching sketch patch ~~S_{ij}~~ from the training set S_{ij} for U_{ij} and (2) feed S_{ij} to VGG-Network and ~~extract~~ compose U_{ij} from the resulting feature maps.

~~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).~~

~~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 boarders 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 boarders of S_{ij} equal to the distances of C_{ij} to boarders 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}.~~

~~Find Matching Sketch Patch~~

~~Estimate Pyramid Column~~ **Find Matching Sketch Patch**

Inspired by previous works [15, 20], we examine the similarity in appearance of photo patches to find a matching sketch S_{ij} for U_{ij}. Denote a patch centered at C_{ij} in the test photo as Ψ_{ij}. If a patch centered at $\tilde{C}_{ij}$ in the kth photo in the training set has the smallest Euclidean distance from Ψ_{ij} in term of color channels, a sketch patch, S_{ij}, centered at $\tilde{C}_{ij}$ is extracted from the sketch paired with the kth photo in the training set. The size and position of S_{ij} are determined by the region associated with U_{ij} which is calculated according to the architecture of the VGG-Network and location of C_{ij}. For non-boarder cells, S_{ij} is a sketch patch of size 144×144 centered at $\tilde{C}_{ij}$. For cells near boarders, the region associated with U_{ij} is the intersection of a sketch patch of size 144×144 centered at C_{ij} with the whole sketch. Therefore, S_{ij} is a patch from the kth sketch containing $\tilde{C}_{ij}$ 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} (see Fig. **??**).

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: $\{G_t^l = A^l \cdot autoref(A^l)^T\}$ where $l \in L_s$.

5.3. ~~Loss Function~~

6. Combine Style with Content

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 i.e.,

$$\mathcal{L}_s autoref(\mathcal{X}) = \sum_{l \in L_s} \frac{1}{M_l^2 N_l^2} autoref \|G^l autoref(\mathcal{X}) - G_t^l\|_2^2 \tag{3}$$

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:

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

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: $\{\hat{G}_t^l = A_c^l \cdot autoref(A_c^l)^T\}$ 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 $\hat{G}^l autoref(\mathcal{X})$ is calculated in exactly the same manner. The component loss is hence defined as:

$$\mathcal{L}_k autoref(\mathcal{X}) = \sum_{l \in L_s} \frac{1}{M_l^2 \hat{N}_l^2} autoref \|\hat{G}^l autoref(\mathcal{X}) - \hat{G}_t^l\|_2^2 \quad (5)$$

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

$$\mathcal{L}_t autoref(\mathcal{X}) = \alpha \mathcal{L}_c + \beta_1 \mathcal{L}_s + \beta_2 \mathcal{L}_k \quad (6)$$

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.

6.1. Implementation Details

Since the VGG-Network is originally designed for color images, while sketches are gray scale images

7. Experiments

In all experiments, we resize test photos and photo-sketch pairs in the training set to a fixed size of 288×288. The final sketch is obtained by resizing the resulting sketch back to the original size. The size of Ψ is 48×48, we modify the first layer of VGG-Network for gray scale images by setting the filter weights to

$$W^k = W_r^k + W_g^k + W_b^k$$

where W_r^k, W_g^k, $\alpha = 0.004$, $\beta_1 = 1$ and W_b^k are weights of the kth filter in the first convolutional layer for the R, G and B channels respectively, and W^k is the weight of the kth filter in the first convolutional layer of our modified network. $\beta_2 = 0.1$. Since generating the final sketch involves an iterative optimization process, the gradient of the final sketch with respect to the content image is not easy to calculate, which prohibits training the content network in an end-to-end manner. The training is therefore carried out by minimizing the squared loss between generated content images and sketches drawn by artists. We evaluate the performance of the proposed method against other state-of-the-art methods on the CUHK student dataset [15] and the AR dataset [10]. For AR dataset, we use the leave-one-out strategy as conducted in previous studies [12, 15]. We compare the results of our method against those of the MRF method [15], weighted MRF (WMRF) method [20], feed-forward CNN (FCNN) method [18] and Filtering based method (SSD) [12].

7.1. Sketch Recognition

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

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

Sketch synthesis methods are usually evaluated quantitatively via the face sketch recognition task [12, 15, 18, 20]. If an algorithm achieves higher sketch recognition rates, it suggests that this method is more effective in synthesizing sketches. We adopt the widely used PCA based recognition method with "rank-1 (R1)", "rank-5 (R5)" and "rank-10 (R10)" criteria [15] where "rank n" measures the rate of the correct answer in the top n best matches. The results of different methods are shown in Table 1. Our method achieves the best performance against all other methods in the "R1" and "R5" tests. There may be two reasons behind this. First, since the test photos are different from those in the training set, we do not expect to find suitable patches from the training set for the target patches. The remedy of linearly combining the patches as in [20] will introduce over smoothing and blurring artifacts, which are also observed in the filtering like approach [12]. Being a generative model, the content network learns the modality transformation locally between photos and sketches, such as transferring edges in photos to strokes in sketches. This helps generating sketches for structures not existing in the training set. Second, our model explicitly minimizes the difference by the style between the synthesized sketches and sketches drawn by artists, while this difference is ignored by all other methods. The performance of all methods in the "R10" test are fairly the same.

8. Final copy

7.1. Style Transfer

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.

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

Table 2. Averaged NGMD value of different methods at different level on AR and CUHK datasets.

~~Please direct any questions to the production editor in charge ofthese proceedings at the IEEE Computer Society Press: Phone (714)821-8380, or Fax (714)761-1784~~where $l \in L_s$. The smaller the NGMD value is, the more similar $\mathcal{X}$ and $\tilde{\mathcal{X}}$ are. Thanks to our style transfer, our results have the smallest NGMD value among all methods under test (see Table 2). It is therefore obvious that our results are the closest to the sketches drawn by artists. Qualitative evaluation results are shown in Fig. **??** and Fig. **??**, where we find that our method provides much clear boundaries and better texture in regions containing hairs and shadings. The results of [18] can roughly outline the sketches, but they do not look like sketches in style. This is also indicated by a high NGMD in Table 2. The results of MRF [15] and WMRF [20] are more similar to hand drawn sketches in style since these exemplar based approaches use patches of hand drawn sketches to make up the target sketch. That is why they have a smaller NGMD value. Inheriting the ability of denoising, the filtering based approach [12] is good at suppressing noises in the results. However, it is also likely for this method to over-smooth the results, which will deteriorate the texture and even introduce blurring artifacts.

7.2. Effectiveness of the model

The loss function we minimize during the generation of sketches contains three terms for content, style and key components respectively. The term $\mathcal{L}_k$ regularizes the results by encouraging the style extracted from the key component regions in the training set to be placed into the key components region of the results, which helps generate better results around these components (see Fig. **??**). To better understand how style influences the final sketch, we smoothly change the emphasis on style by adjusting β_1 and β_2 while keeping α fixed. Fig. **??** indicates that the sketch with style transfered contains more texture and is more like a drawn sketch. The Theano implementation of the proposed method takes approximately 100 seconds to generate a sketch on a GeForce GTX TITAN X platform. The bottle neck lies in the style transfer which requires feeding $\mathcal{X}$ to the VGG-Network to estimate targeting feature maps and to calculate the gradient of Eq. (6), which is computationally intensive.

8. Conclusion and Future Work

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

References

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

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

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

[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, 3, 4, 5

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

[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] A. Martinez. R. benavente. the AR face database. Technical report, CVC Tech. Report, 1998. 6

[11] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2, 3, 4

[12] 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

[13] 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

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

[15] 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

[16] 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

[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

[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

[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, 5, 6, 7