Stylized Sketch Generation using Convolutional Networks

Mayur Hemani
Adobe Inc.
Adobe Systems India Pvt. Ltd.
A-05, Sector-132
Noida, U.P.-201304, India
mayur@adobe.com

Abhishek Sinha
Adobe Inc.
Adobe Systems India Pvt. Ltd.
A-05, Sector-132
Noida, U.P.-201304, India
abhsinha@adobe.com

Balaji Krishnamurthy
Adobe Inc.
Adobe Systems India Pvt. Ltd.
A-05, Sector-132
Noida, U.P.-201304, India
kbalaji@adobe.com

ABSTRACT

The task of synthesizing sketches from photographs has been pursued with image processing methods and supervised learning based approaches. The former lack flexibility and the latter require large quantities of ground-truth data which is hard to obtain because of the manual effort required. We present a convolutional neural network based framework for sketch generation that does not require ground-truth data for training and produces various styles of sketches. The method combines simple analytic loss functions that correspond to characteristics of the sketch. The network is trained on and evaluated for human face images. Several stylized variations of sketches are obtained by varying the parameters of the loss functions. The paper also discusses the implicit abstraction afforded by the deep convolutional network approach which results in high quality sketch output.

Keywords
neural-networks, image manipulation, image stylization, sketch style

1 INTRODUCTION

Stylized sketch synthesis from a photograph is an important problem in image stylization. In this paper we propose a simple framework for stylized sketch generation from images using a convolutional neural network used as an image-to-image optimization framework. We present results and analysis for human face images.
Sketches are representational artwork characterized by minimal marks (strokes, shading, etc.) on paper or other substrates. They can be hand-drawn or digitally constructed using software, and in this work we limit our scope to digitally styled sketches only. These sketches usually start from a photographic image, and transform it by means of an image processing pipeline.

Static filter pipelines (for example in [11]) are a fast method for producing sketch-like effects. The drawback of such an approach is that its ability to capture features of a sketch drawing depends on complex heuristics that do not always work. Also, these methods lack control over the characteristics of the output sketch which depends heavily on the input image. Our method addresses both of these concerns by learning a function from image data using an optimization objective that captures the semantics of the desired sketch output. Different stylized sketch output can be constructed simply by changing the parameters and structure of the loss functions of the underlying neural network.
Neural networks are now commonly used for image stylization, starting with [6]. Sketch generation could be posed as a style-transfer problem. The primary drawback of this approach is that examples images of each sketch style are required. Our method addresses

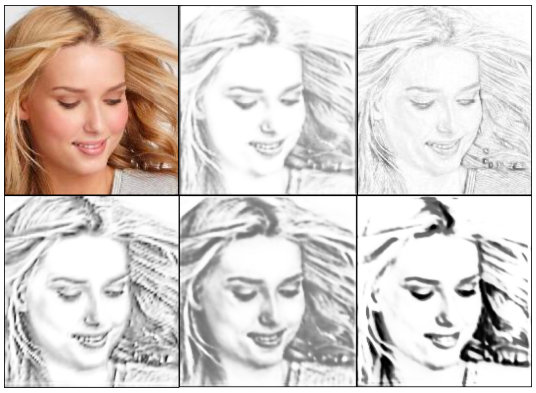
Figure 1: Sketching results using our method. The different styles of sketches are produced by varying the parameters and structure of the loss function at the top of an image-to-image neural network.

this problem by not requiring any specifically collected paired-data. Also, the results vary significantly with style-transfer.
Stylized sketch generation can also be set up as a supervised learning problem, but the data for stylized sketches is scarce. Generative approaches using adversarial training, such as in [20], have also been proposed but they too lack control over the characteristics of the sketch output.

Our method for photograph to sketch generation comprises of a U-Net [12] like feed-forward convolutional neural-network which is trained on different loss functions that capture the semantics of the sketch output - its similarity to the input photograph, its overall brightness and contrast, and its sharpness, as determined by the amount of variation captured. We fuse a set of non-trainable classical image-processing filters at the top of the convolutional network that enables the styles to be varied either by direct combination with the output, or by enforcing a semantic constraint on it. Different loss function parameters produce different stylizations. The data used for training is a set of regular, non-sketch photographs. The network trains quickly, for fewer than ten thousand iterations, and produces high quality sketch output.

This paper makes the following contributions:

- A generalized method for fusing classical image processing techniques into a learnable filter mechanism as constraints.
- Loss functions for producing stylized sketch output, that are also extensible to other kinds of image stylization tasks.

2 RELATED WORK

Our work on sketch generation is closest to those based on style-transfer because both methods use end-to-end training of neural networks without the need for extensive data. Supervised learning methods like [1, 14, 13, 21, 18, 25, 26, 24, 19, 17, 4] learn from large sets of paired data and are directed towards producing hand-drawn sketches, and are therefore not directly related to our work.

Style Transfer Neural Style Transfer methods like [6], [8] are closely related to our work because they too train a deep neural network without the use of paired data. [8] extended the initial algorithm in [6] by training the image transformer network for each style image, thus making it possible to generate stylized images for a single style in a single pass. [2] proposed to use conditional instance normalization in image transformer networks. Their method reduces each style to a point in the embedding space and makes it possible to train a single network for several different styles. Recently, [3] obviated the need to retrain the image transformer networks for every new style by proposing a meta network that takes a style image and produces the transformer network directly. All of these methods rely on matching the low- and mid-level features of deep neural network for transferring style from a style source to another image. Our method differs from style transfer in that the style source is itself expressed as semantic constraints on the output, rather than another image.

Deep Image Prior An important reason why the method is able to produce stylized sketch images with ease is that the underlying neural network affords it a certain level of abstraction of the representational qualities of the input images. This ability of the network is discussed in the work by Ulyanov et al. [15]. They discuss how a minimally trained U-Net network ([12]) provides a good prior for the input itself and impedes noise (details). This can be considered a means of implicit abstraction of photographic images. We use this impedance of the network to noise and detail to the effect of producing a pre-styled, abstract state. The constraints imposed by the filters then enable directing the output towards the desired sketch characteristics.

3 METHOD

An overview of our framework is depicted in Figure 2. The learning pipeline comprises of a U-Net like convolutional neural network with a set of fixed, non-trainable filters at the top. The network learns to optimize a composite loss function which is constructed on the input image, the raw unfiltered output from the network and the filtered output. The sketch output is either a filtered output from the fixed filters or a linear combination of the raw unfiltered and filtered outputs. Combinations of different fixed filters and different loss functions lead to distinct sketch stylizations. For contrast independence, we employ the instance normalization trick from [16].

3.1 Dataset

The training of our network does not require any ground-truth data. We trained and validated our framework on human face images as input. We use the Celeb-A [10] aligned and cropped images for our experiments. The dataset has over 200,000 images of human faces. We use around 30,000 of these images for training, and the rest for testing. The images are center-cropped to a size 192x192 pixels for training. We also validate our method with images from the CUHK Face Sketch Database [22].

3.2 Network Architecture

The network is a U-Net [12] like fully convolutional neural network. We chose the U-Net like architecture

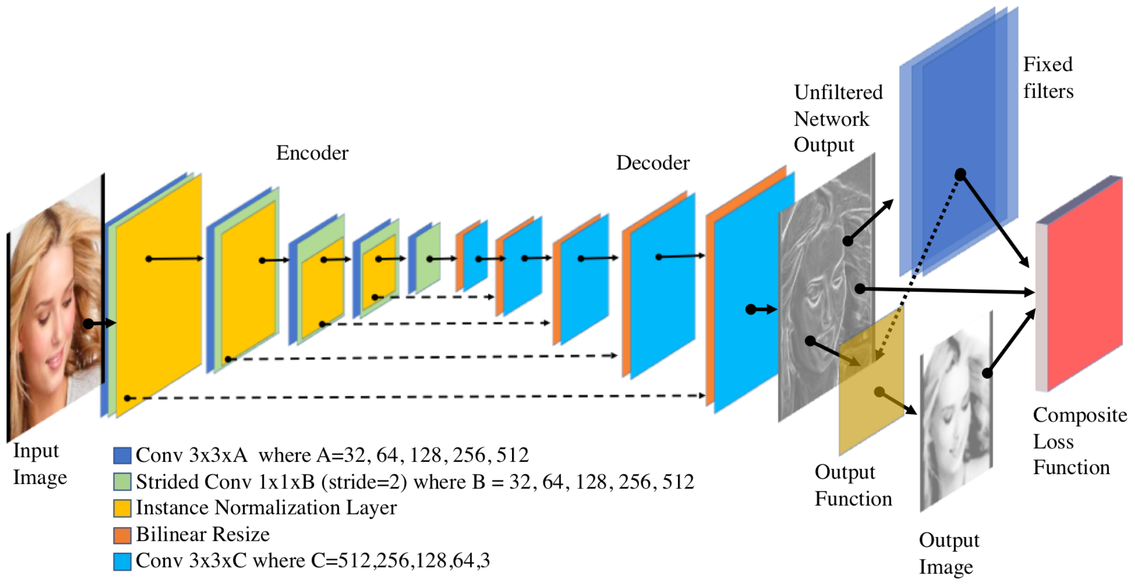
Encoder
Decoder
Unfiltered Network Output
Fixed filters
Input Image
Composite Loss Function
Output Function
Output Image
🟦 Conv 3x3xA where A=32, 64, 128, 256, 512
🟩 Strided Conv 1x1xB (stride=2) where B = 32, 64, 128, 256, 512
🟨 Instance Normalization Layer
🟧 Bilinear Resize
🟦 Conv 3x3xC where C=512,256,128,64,3

Figure 2: Network architecture.

because of the implicit prior that it affords, as discussed in [15]. In our experiments, this network trained really quickly as long as the expected output as close to the input image itself, the closeness measure itself being codified by the loss functions. All training sessions were run for fewer than ten thousand iterations. The last layer of the network has a sigmoid activation and produces a 3-channel image output. This output is then fed into a set of fixed filters including edge detection, and blur filters. The resulting output image, as well as the raw, unfiltered output from the network, along with the input images are all passed into the composite loss function. The output is selected as either the filtered output, or a combination of the filtered and unfiltered output images.

3.3 Loss Formulation

We consider sketches as an interpolate between the grayscale image and the inverted edge response image, along with some modified textural characteristics. This transformation is achieved by means of the neural network working as an image-to-image filter that is trained to optimize certain functions of the output image, and satisfy constraints defined on classical image processing filters over the output. For example, pencil or charcoal sketches can be considered as gray-scale image representations of a photographic image which have the following properties:

- The shapes and edges of the sketch match with those of the photographic image.
- The substrate color (white for the paper, for instance) is higher in mass than the pencil/charcoal marks.

Output Characteristic	Loss Representations (Mean)
Brightness Similarity	$L_s = \|\|G(Y) - G(X)\|\|^2$
Edge Similarity	$L_e = \|\|\nabla Y - \nabla X\|\|^2$
Overall Brightness	$L_b = \|\|G(Y)\|\|$
Local Brightness	$L_{lb} = \|\|\mu_{n\times n}(G(Y)) - \mu_{n\times n}(G(X))\|\|^2$
Overall Contrast	$L_c = \|G(Y)\|_\infty^{n\times n} + \|\text{-}G(Y)\|_\infty^{n\times n}$
Total Variation	$L_{tv} = \|\|\nabla Y\|\|$

Table 1: Key image characteristics used for constructing loss functions. X is the input image, Y is the output, $G(x)$ is the grayscale conversion of x, ∇ is the gradient operator, $\mu_{n\times n}$ is the $n \times n$ average pooling operation, $||_\infty^{n\times n}$ is $n \times n$ max-pooling.

These ideas are used to formulate loss functions that produce different variants of sketch outputs. The general formulation includes loss terms for edge similarity L_e, brightness similarity L_{lb}, overall brightness L_b and contrast L_c of the output, its closeness to a binary image F_{bin}, a function of the overall brightness of the image (Luma CCIR601) that determines the degree of whiteness in the output F_{mass}, and a total variation term L_{tv}. The similarity terms are measured with respect to the original image. More explicitly, this is expressed as:

$$\begin{aligned} L(X,Y) =& \lambda_{bdif} L_s + \lambda_{edgedif} L_e + \lambda_{lbdif} L_{lb} \\ &+ \lambda_{bright} F_{mass}(G(Y)) + \lambda_{contr} L_c \\ &+ \lambda_{bin} F_{bin}(G(Y)) + \lambda_{var} L_{tv} \end{aligned} \tag{1}$$

Here, X is the RGB color input image, Y is the RGB output image, $G(Y)$ is the grayscale image corresponding to the output image and the λs are constant weights. The different variations of sketch output are produced

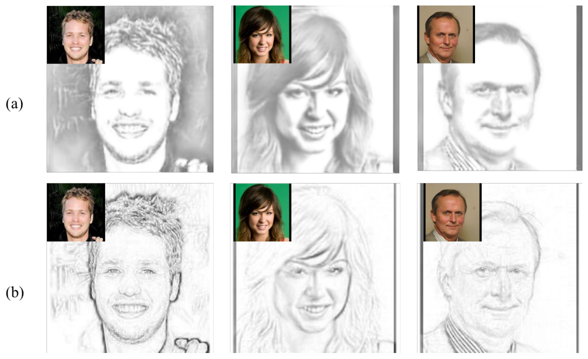

Figure 3: Pencil Sketch Output.

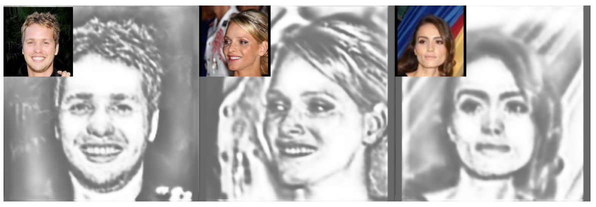
Figure 4: Charcoal sketch output.

by choosing different values for the λs, F_{mass}, F_{bin}, and the implementation of the gradient (∇) operator. Table 1 lists the different loss function components.

3.3.1 Shaded Pencil Sketch

Shaded pencil sketches (Figure 3(a)) are characterized by thin strokes and smears over the paper. The constants are $\lambda_{edgedif} = 1.0$, $\lambda_{bright} = 10^{-2}$, $\lambda_{bin} = 10^{-3}$.. The loss function components are:

$$\begin{aligned} L_e =& ||\nabla Y - \nabla X||^2 \\ F_{mass}(x) =& (1.0 - G(x) \cdot x) \\ F_{bin}(x) =& ||(1.0 - 4(x - 0.5)^2)|| \end{aligned} \tag{2}$$

3.3.2 Traced Pencil Sketch Output

A contoured sketch output (Figure 3(b)) can be produced where instead of edge-matching with the Sobel operator, for the input image X, we employ a formulation which is similar to the Difference of Gaussians method for edge detection as used in [23] and implementing a selection operation on the edges in the input image. The image predicted by the network (Y), is a selection mask over the edges. The objective function itself is to minimize the difference between the Y_{out} image and the grayscale vector of the input image $G(X)$, as well as reduce the overall edge response of the predicted mask Y.

$$\begin{aligned} G_{pooled} =& |G(X)|_{\infty}^{4\times4} \\ G_{shifted} =& R_{4\times4}(G_{pooled}, w, h) \\ \nabla X =& G_{shifted} - G(X) \end{aligned} \tag{3}$$

Where $R_{4\times4}$ is a re-scaling operation that restores the size of the max-pooled image to the original size. The output image is computed as:

$$Y_{out} = 1.0 - \nabla G(X) \cdot G(Y) \tag{4}$$

$$L(X,Y) = \lambda_{lbdif} ||Y_{out} - G(X)||^2 + \lambda_{tv} L_{tv} \tag{5}$$

Where $\lambda_{lbdif} = 1.0$, and $\lambda_{tv} = 0.01$

3.3.3 Charcoal Drawing Generation

Charcoal drawings (Figure 4) have larger smears and deeper black shades. The corresponding loss function components are the same as for shaded pencil sketches 2 except for the additional contrast terms and the λ values:

$$L_c = |G(Y)|_{\infty}^{4\times4} + |-G(Y)|_{\infty}^{4\times4} \tag{6}$$

with $\lambda_{edgedif} = 1.0$, $\lambda_{bright} = 10^{-3}$, $\lambda_{contr} = 0.1$, and $\lambda_{bin} = 0.1$.

3.4 Contrast Invariant Output

One major reason why we employ a learning method for sketch generation is that it can be used to learn the map from a color image to a sketch image irrespective of the contrast of the input. For this, instance-normalization (as described in [16]) layers are inserted into the encoder part of the network. Figure 5 shows the results without and with the instance normalization trick.

3.5 Results on other datasets

Figure 6 depicts some results from the CUHK Face Database. These results are not really comparable to the hand-drawn ground-truth images, but they are produced for comparison with other contemporary methods.

3.6 Training and Stability

We use the Adam optimizer [9] for training the network. Many of the loss functions we use have stable local minima which poses a challenge in training. For example, if the network is initialized randomly, and trained with L_{bin} as part of the loss function, sometimes it results in an inverted initial output. And then it takes very long for the network to converge towards the global optimum (or even a good local optimum). We could reduce the learning rate or reduce the weights of these losses considerably, but that too results in delay in convergence. We found that training with the L_2 loss between the input (grayscale) and output images (reconstruction task) for only a thousand iterations initially, produces a very good prior and adds to the stability of

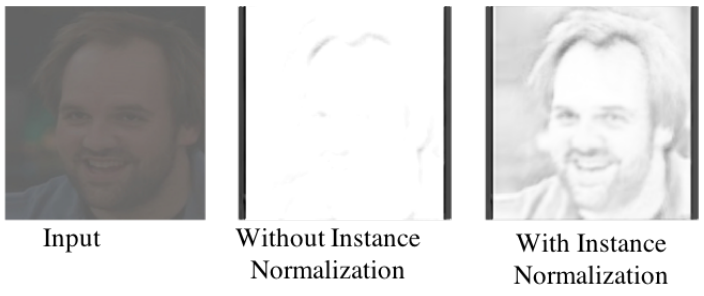

Figure 5: Output without and with instance normalization.

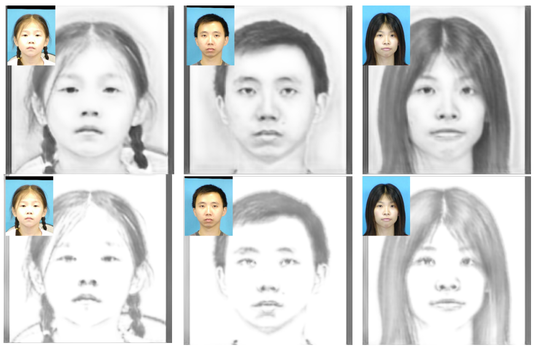

Figure 6: Pencil and charcoal sketch output for the CUHK dataset. Top row: Charcoal sketches, Bottom row: pencil sketch output.

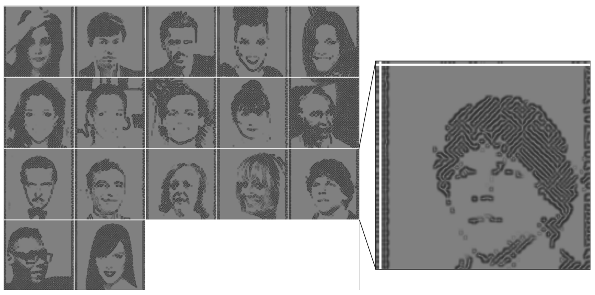

Figure 7: Quilt-like sketch output produced by matching the inverted edge image against the grayscale input image.

the training. In our experiments, with an initial training for reconstruction, the network never converged to a stable local optimum.

Just as in illustration workflows, where an abstract shape or form serves as a good starting point for producing styled art, this abstraction implicit in the network serves as a good starting point for learning automatic stylization. We can use this fact to produce artistic renditions of the input image that abstract away certain representational qualities but retain the semantics. Sketch generation is an instance of this sort of abstraction.

To further corroborate the insight that the network does learn a representation close to the original image even when it is not fully trained, we train our network with a combination of three conflicting loss function with clear trade-offs. This ensures that the network never fully converged to the minimum of either of the loss function components:

$$\begin{aligned} L(X,Y) &= L_{edge} + L_{bdif} \\ L_{edge}(X,Y) &= ||(1 - F((G(Y)) - G(X)||_2 \\ L_{bdif}(X,Y) &= ||\mu_{4\times 4}(G(Y)) - \mu_{4\times 4}(G(X))||_2 \end{aligned} \tag{7}$$

where the filter function F(Y) is the average of the gradient responses (computed using the Sobel operator) in the x- and y- directions.

$$F(Y) = 0.5 \cdot (\nabla_x(Y) + \nabla_y(Y)) \tag{8}$$

The loss functions pit the (inverted) edge response against the grayscale value of the input image. For photographic images, the gradient response is rarely equal to the input image G(X), because the intensities of the pixels are not exponential in their coordinates. Therefore, this is a very hard optimization problem for the network. As a result, the network learns to abstract out details and produces only the most necessary edge responses. Training with this loss function produces a quilt-like pattern(Figure 7) from the network.

3.7 Comparison with Style Transfer

Our results are compared with those from style transfer methods. To produce results from style transfer, we took a synthesized sketch image from our method as a style image and used it to transfer style over to a photograph in the dataset via the approach of neural style transfer as in [6]. The results are depicted in Figure 8.

3.8 Extended Results

The versatility of our approach is indicated by the diversity of results that can be obtained by simply changing the underlying loss function parameters. Figure 9 demonstrates the results on high-resolution images. The last output is obtained by capturing the color output times the inverted edge response, while keeping identical objectives as a pencil sketch, besides an L_2 distance between the color output and the input image.

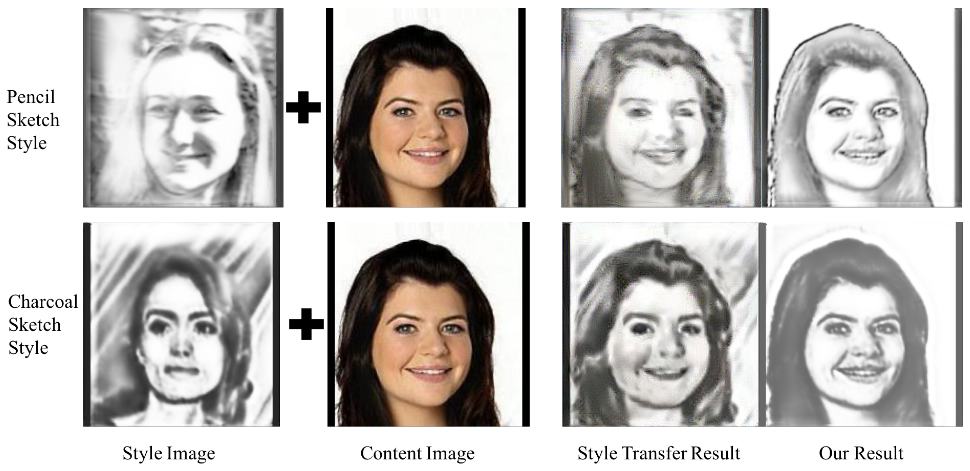

Figure 8: Comparison with style transfer outputs. The style transfer output loses semblance with the content image, while our method produces better looking sketch output.

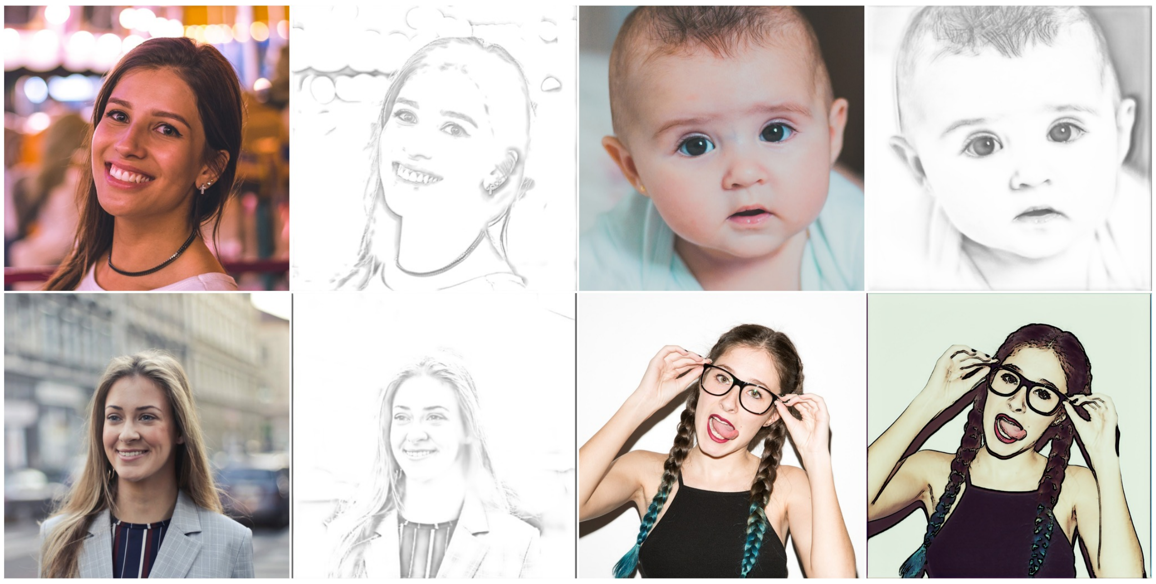

Figure 9: High Resolution Results. The last result is obtained by capturing the color output from the network times the inverted edge response as the output.

4 CONCLUSIONS

This paper presents a framework for producing stylized images without the need to train with paired style and ground-truth data. It focuses on producing stylized sketch output by means of fusing classical image processing filters into a learning framework based on an intuitive loss function. It also demonstrates how the underlying abstraction afforded by the deep neural network aids in producing stylizations.

5 REFERENCES

[1] Chaofeng Chen, Xiao Tan, and Kwan-Yee K. Wong. Face sketch synthesis with style transfer using pyramid column feature. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 485–493. IEEE, 2018.

[2] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. 2017.

[3] Shuicheng Yan Falong Shen and Gang Zeng. Neural style transfer via meta networks. In *CVPR2018*, 2018.

[4] Fei Gao, Shengjie Shi, Jun Yu, and Qingming Huang. Composition-aided sketch-realistic portrait generation. *arXiv preprint arXiv:1712.00899*, 2017.

[5] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style. *CoRR*, abs/1508.06576, 2015.

[6] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional

neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, 2016.

[7] Xun Huang and Serge J. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. *CoRR*, abs/1703.06868, 2017.

[8] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 2016.

[9] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[10] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[11] Cewu Lu, Li Xu, and Jiaya Jia. Combining sketch and tone for pencil drawing production. In *Proceedings of the Symposium on Non-Photorealistic Animation and Rendering*, NPAR '12, pages 65–73, Goslar Germany, Germany, 2012. Eurographics Association.

[12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[13] Yibing Song, Linchao Bao, Qingxiong Yang, and Ming-Hsuan Yang. Real-time exemplar-based face sketch synthesis. In *European Conference on Computer Vision*, pages 800–813. Springer, 2014.

[14] Yibing Song, Jiawei Zhang, Linchao Bao, and Qingxiong Yang. Fast preprocessing for robust face sketch synthesis. *arXiv preprint arXiv:1708.00224*, 2017.

[15] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. *arXiv:1711.10925*, 2017.

[16] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *CoRR*, abs/1607.08022, 2016.

[17] Lidan Wang, Vishwanath Sindagi, and Vishal Patel. High-quality facial photo-sketch synthesis using multi-adversarial networks. In *Automatic Face & Gesture Recognition (FG 2018), 2018 13th IEEE International Conference on*, pages 83–90. IEEE, 2018.

[18] Nannan Wang, Dacheng Tao, Xinbo Gao, Xuelong Li, and Jie Li. Transductive face sketch-photo synthesis. *IEEE transactions on neural networks and learning systems*, 24(9):1364–1376, 2013.

[19] Nannan Wang, Wenjin Zha, Jie Li, and Xinbo Gao. Back projection: an effective postprocessing method for GAN-based face sketch synthesis. *Pattern Recognition Letters*, 107:59–65, 2018.

[20] Nannan Wang, Mingrui Zhu, Jie Li, Bin Song, and Zan Li. Data-driven vs. model-driven: Fast face sketch synthesis. *Neurocomputing*, 257:214–221, 2017.

[21] Shenlong Wang, Lei Zhang, Yan Liang, and Quan Pan. Semi-coupled dictionary learning with applications to image super-resolution and photo-sketch synthesis. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2216–2223. IEEE, 2012.

[22] X. Wang and X. Tang. Face photo-sketch synthesis and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(11):1955–1967, Nov 2009.

[23] Holger Winnemöller, Jan Eric Kyprianidis, and Sven C. Olsen. Xdog: an extended difference-of-gaussians compendium including advanced image stylization. *Computers & Graphics*, 36(6):740–753, 2012.

[24] Dongyu Zhang, Liang Lin, Tianshui Chen, Xian Wu, Wenwei Tan, and Ebroul Izquierdo. Content-adaptive sketch portrait generation by decompositional representation learning. *learning*, 2:111.

[25] Shengchuan Zhang, Xinbo Gao, Nannan Wang, and Jie Li. Robust face sketch style synthesis. *IEEE Transactions on Image Processing*, 25(1):220–232, 2016.

[26] Shengchuan Zhang, Xinbo Gao, Nannan Wang, Jie Li, and Mingjin Zhang. Face sketch synthesis via sparse representation-based greedy search. *IEEE transactions on image processing*, 24(8):2466–2477, 2015.