Learn to Sketch: A fast approach for universal photo sketch

Zhi-Song Liu[1], Wan-Chi Siu[1, 2], and H. Anthony Chan[1]

[1]Caritas Institute of Higher Education, Hong Kong
[2]The Hong Kong Polytechnic University, Hong Kong

***Abstract*—Rendering real photos to abstract sketches is an interesting application that can help us understand the key features. In this paper, we propose a universal photo sketch model via a deep convolutional neural network. Prior arts often cast this problem as an edge or contour detection. However, the edges or contours may not exactly reflect the boundaries of the contents of the photos. They also fail to reveal the occlusion that separates the objects from each other. We resolve this problem by proposing Photo2Sketch and Sketch2Photo to form a loop to bridge the gap between photos and sketches. We introduce relevant sketch references as indicators to supervise the sketch generation. Meanwhile, we also introduce an adaptive sketching process that can generate drawing with confidence, hence multiple sketches can be obtained. Experimental results show that our proposed method surpasses other state-of-the-art methods in both qualitative and quantitative measures.**

I. Introduction

In computer vision or computer graphics, edge-like visualization is one of the interesting tools in image representation. It can be edges, textures, contours, boundaries, and so on. Using edge-like visualization can reduce the data redundancy in image processing. Automatic edge-like generation can be useful for geometric understanding [7], object recognition [9], fashion design [14] and other applications. Sketch, in a general definition, is one type of edge representation of the image. Yet it is not just edges but connects related edges to form continuum boundaries for objects. It is not even just contours but preserves more detailed features to reveal the occlusion of objects. It differs from semantic segmentation, sketches can describe more detailed textures. Perceptually, it resembles how human observe the world. For art design, a sketch is an important artistic language that can reflect the rationality of visual perception and aesthetic appreciation.

However, many related works approximate sketch as edge-like visualization [3]-[19]. For example, [4] gives an edge detection model that extracts multiscale holistic feature representation to find edges that are close to the object boundaries. [6] forms a new dataset with paired photos and sketches and uses it to train a deep network for sketch generation. Despite the existing sketch datasets, it still cannot cover the large variety of different images with different contents. [15] instead, proposes a deep model that works as a domain transfer. As an unsupervised approach, it can mimic the appearance of the prediction close to the sketch references. However, it only works well on specific images and loses a lot of details when it is applied to image with new contents.

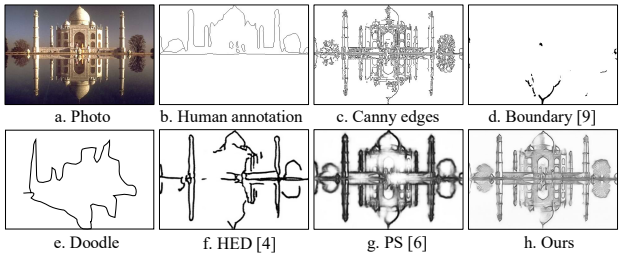

Fig. 1: **Comparison among different edge-like generation.** We use one image from BSD500 [27] and compute it using different edge/contour detectors. Our result shows superior visual quality compared to others in details.

In order to make attractive sketch generation, we propose in this paper Photo2Sketch by means of projection from photos to sketches. Having consulted experts on art painting, we define our objectives, which differ from previous studies, in the following: 1) the sketch results should contain edges and contours that can separate objects from each other and reveal the occlusion relationship, 2) the sketch results should also preserve the depth information and complex textures by the density of the drawings, and 3) the sketch results should have some drawing uncertainty reflected by the color of the drawing lines. These objectives ensure that our generated sketch image can be perceptually close to the human sketch.

As shown in Figure 1, we compare with the results of 1) simple edge detection using canny detector (Figure 1c), 2) Boundary detection (Figure 1d), 3) free doodle (Figure 1e), 4) Holistic edge detection (Figure 1f) and 5) contour and edge detection (Figure 1g). It can be found that our result outperforms others in visual details. For instance, our result can keep the windows of the building and the shadow in the water. We use Figure 2 to demonstrate the desired sketch image that we target. By using our proposed method, the facial details are preserved and the style looks like human drawing. To summarize, our key contributions include:

- We implicitly discover mapping correlation between photos and sketches by forming a loop to achieve self supervision. That is, we use Photo2Sketch to map photos to corresponding sketches and use Sketch2Photo to map sketches back to photos.
- To generate photo-realistic sketches, we introduce ref-

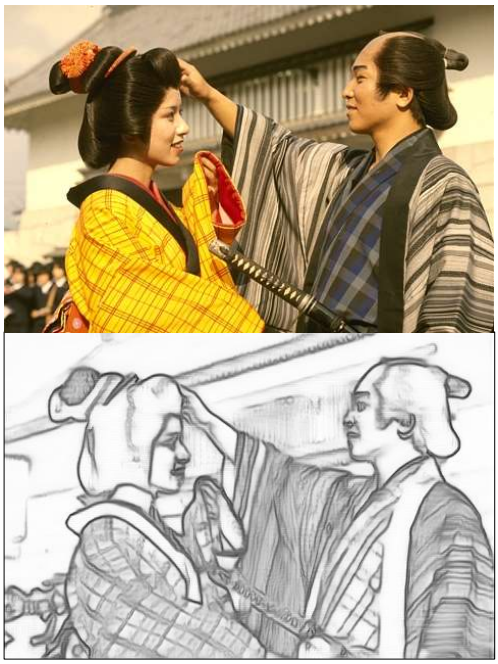

Fig. 2: **Example of using our Photo2Sketch for sketch.**.

erence based training losses to encourage the generated sketches close to general sketches.

- Furthermore, we introduce the soft weighted function to adaptively generate the sketch results so that we can assign confidences to different drawing lines. Hence, multiple sketch candidates can be generated.

II. Related Work

In this section, we will give a detailed introduction to the previous works, including 1) edge based image generation and 2) Doodle based image generation.

A. Edge based image generation

We further categorize edge based image generation into three groups: 1) edge detection, 2) contour detection, 3) category-aware edge detection and 4) thinning and skeleton detection.

Edge detection traditionally use first-order or higher-order filters to find horizontal and vertical edges [1-2]. However, these traditional approaches are sensitive to noise and sudden intensity changes within the local regions. The generated edges are redundant and noisy. Holistic edge detection [4] is proposed by using a deep neural network to extract rich feature representation to find the critical edges and boundaries that can reflect the 3D geometry. Similar techniques are further developed in [3]-[5]. For example, [3] makes use of multiple edge detectors to form soft labels to supervise the coarse-to-fine training process. [5] further refines edge detection as close as possible to the ground truth with subpixel convolution. Along with many proposed edge detection approaches, there are some labeled datasets with paired photos and edge maps, like BSD500 [27] and NYUDv2 [7], which can be used for model training and evaluation.

For contour detection, it serves as the basic computer vision task such as image segmentation. The idea is to classify pixels into different classes and then find the boundaries that partition objects. [8] classifies the contours into sub-classes of positive and negative samples. It then uses the positive-sharing loss function to regularize the model training. [9] proposes a fully convolutional encoder-decoder network to extract multi-scale features for contour detection. To generate more contour samples, [10] proposes a technique that generates weakly supervised annotations with more accurate boundaries. Category-aware edge detection is a more specific contour detection task that objects with the same classes should be assigned with the same contours. It is a dual task to semantic image segmentation which identifies object regions. For example, [11] proposes a semantic boundary thinning layer to reduce boundary ambiguity. It also uses an active alignment scheme to iteratively refine the ground-truth labels by prediction. To localize the contours for smaller objects, [12] enhances the contours detection using their proposed dynamic feature fusion.

For thinning or skeleton detection, it is complementary to contour and edge detection. It is widely used in shape-based object matching and recognition. The idea is to find the *symmetric axis* that can represent the structure, presence and size of the objects. In the early works, [37-39] proposed morphological image transforms to achieve fast thinning detection. More recent studies [40-43] show that using deep learning approaches can achieve good performance both in running time and detection accuracy.

B. Doodle based image generation

Doodle based image generation is an active research topic. It is close to edge-based image generation to extract edges from the photos. The difference is that it extracts more abstract and simple edges to mimic kids' drawing. For example, [13] proposes a recurrent neural network (RNN) able to generate simple drawings from images. However, it cannot process images with complex objects. [14] proposes a deep generative model for generating high-quality multiclass sketches via conditional coding, hence it can generate the desired sketches. To learn non-local properties, [15] proposes a transformer based sketcher which learns the optimal stroke sequencing strategies that generate the most recognizable and distinguishable strokes. DoodlerGAN [16] was proposed to achieve interactive drawing. The idea is to use a part-based Generative Adversarial Network (GAN) to fill out missing parts of the object. Another popular topic is face based doodle generation. It focuses solely on facial images to preserve facial features. [17] proposes a cycleGAN structure to form a close loop between unpaired photos and sketches to learn the mapping functions. [18] uses robotic arms to draw stylized avatar to form unpaired faces and avatar images to train a GAN network for doodle generation. [19] proposes a pencil stroke imitation mechanism that can guide stroke drawing by a novel edge tangent flow field.

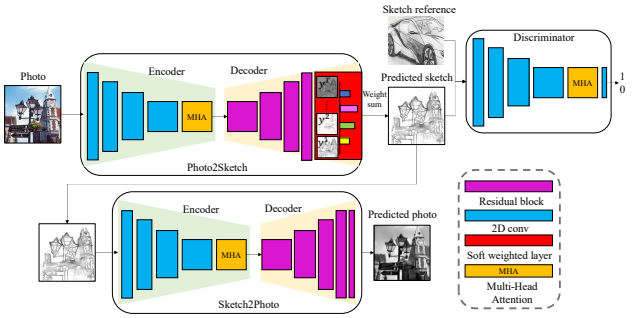

Fig. 3: **The training architecture of the proposed method.** It contains three sub-networks, Photo2Sketch, Sketch2Photo and discriminator. The Photo2Sketch and Sketch2Photo are both built on the same encoder-decoder structure. The Encoder and decoder are symmetric to each other to down- and up-sample feature maps 8×. The Discriminator is a simple 2D CNN structure that down-sample the feature maps 8 × to output a probability map. **During the testing, we only keep the Photo2Sketch for sketch generation.**

Further Highlights To distinguish our proposed method from other existing edge based or doodle based image generation, let us summarize the differences in the following points: 1) we propose a sketch generator that can transfer photos to sketch-like images with edges and detailed textures, rather than simple edges or contours [8]-[12], 2) our proposed Photo2Sketch can preserve the 3D geometry to reveal the original contents rather than abstract doodle [13]-[19], and 3) rather than focusing on images with specific classes [17]-[19], the proposed method is general for photos with any contents.

III. Method

In this section, we give a detailed introduction to our proposed Photo2Sketch and Sketch2Photo. It has two major advantages: First, it can learn the forward mapping function to cast photos to sketches, and it can also learn the backward mapping function to project sketches to photos. Second, it can generate sketches with different confidence so we can obtain multiple sketches with different details.

Overview. Given an input image **X** and a style (reference) image **R**, the goal of our proposed Photo2Sketch is to find the mapping model $g : \mathbf{X}|\mathbf{R} \rightarrow \mathbf{Y}$ to map the input photo to a sketch prediction with the style transferred from the style image. Thus, we can obtain the sketch prediction **Y** with contents close to **X** and style close to **R**. Photo2Sketch is a feed-forward network that transfers arbitrary photos efficiently (~100 fps, Figure 3). It consists of three sub-networks: Photo2Sketch, Sketch2Photo and discriminator. The Photo2Sketch and Sketch2Photo are both built on the same encoder-decoder structure. The Encoder and decoder are symmetric to each other to down- and up-sample feature maps 8×. At the bottleneck of the encoder, we have one Multi-Head Attention (MHA) that learns the global feature correlations. For the last layer of the decoder of Photo2Sketch, we propose a soft weighted layer to learn the sketch image. The Discriminator is a simple 2D CNN structure that down-samples the feature maps 8 × to output a probability map. Let us introduce them in details.

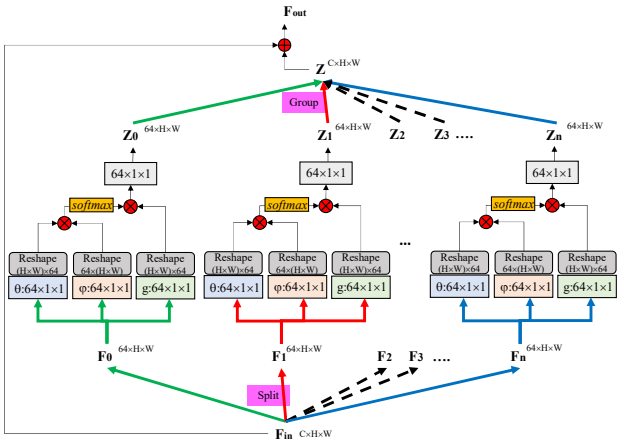

Fig. 4: **The structure of Multi-Head Attention.** We show the computation of MHA that split input features into n heads and compute self-attention in parallel, then it groups heads together and add the input feature to the final output.

A. Photo2Sketch

The Photo2Sketch and Sketch2Photo are identical to each other with different functions. They are built upon a symmetric encoder-decoder structure. The encoder is to learn compact feature representation and the decoder is to cast the learned features to the target domain, that is, to produce sketches for Photo2Sketch and to form photos for Sketch2Photo. The encoder is made of 4 layers of 2D convolution layers (blues boxes in Figure 3) and one Multi-Head Attention (MHA) block. The use of MHA block is inspired by recent studies on non-local processing in deep learning [20]-[22]. The advantage is that it can learn global feature correlations to better grasp the image information. It also fits our target to sketch generation. Using MHA can extract the global geometric information of the input photo so it can preserve the 3D geometry for reconstruction. Note that the structure of MHA follows the design of [21] to efficiently extract global attention. In Figure 4, we show the structure of the MHA block. It splits the input feature F_{in} into n heads, and each head has a feature map of size $64 \times H \times W$. Each head computes its self-attention, and all heads are regrouped together and added with the input feature to form the output feature F_{out}.

For the Decoder, we make use of four layers of residual blocks to gradually upsample the feature maps back to the same dimension as the input photo. Recall that our target is to generate a black-white sketch to reflect the human drawing so it is natural to output a grayscale image for reconstruction [3]-[6]. Instead, we propose to use a soft weighted structure (the red box in Figure 3) to construct the sketch image. The idea is that we can consider the sketch generation as a combination of multiple sketch predictions that are weighted by confident scores. Hence, the soft weighted function can be written as,

$$y_{i,j} = \sum_{t=0}^{T} \frac{e^{y^t_{i,j}}}{\sum_{t=0}^{T} e^{y^t_{i,j}}} \cdot y^t_{i,j} \tag{1}$$

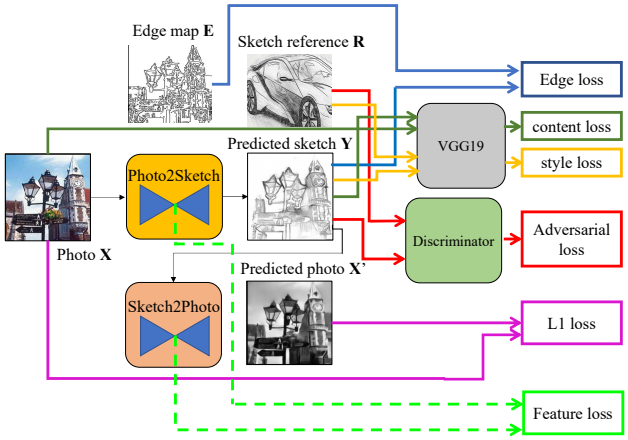

Fig. 5: **The training losses of the proposed method.** To train the proposed network, we use 6 loss terms, including Edge loss (edge map predicted sketch), content loss (predicted sketch and photo), style loss and adversarial loss (sketch reference predicted sketch), L1 loss (photo and predicted photo) and feature loss (encoder features from Photo2Sketch and Sketch2Photo).

where T is the number of candidates (also can be seen from the red box in Figure 3), $y_{i,j}$ is the i,j-th predicted pixel on the final sketch image and $y^t_{i,j}$ is the i,j-th feature value of the t-th predicted sketch image. This soft weighted function is differentiable so it can be trained end-to-end. There are T different sketch results that are weighted together to obtain the final sketch image. To adaptively generate multiple sketches, we can randomly select the t-th sketch image.

B. Sketch2Photo

The use of Sketch2Photo is to learn the inverse mapping function from sketches to photos. The structure of Sketch2Photo is as same as Photo2Sketch. It resonates with the studies on CycleGAN [23]. The reason for using Sketch2Photo is to achieve self-supervision. Since we do not have paired photo-sketch dataset, we can learn one more step to map the predicted sketch back to the original photo. Note that we do not care about the color distribution so we only design the Sketch2Photo to generate grayscale photos.

C. Discriminator

The discriminator is built using 4 layers of 2D convolution followed by a MHA block. It is used to output a probability map to classify the predicted sketch from the reference. The objective is to encourage the predicted sketch close to the reference in appearance. We also use a MHA block to grasp the global information.

D. Training strategy

With the proposed network, we train this network to generate sketches with 1) accurate edges and contours and 2) detailed textures. The complete training process is described in Figure 5

There are 6 different loss functions to guide the sketch generation. First, we have the **edge loss** L_{edge} with l_1 errors between the predicted sketch Y and edge map E. The Edge map is obtained by using the canny detector on the input photo. Note that we use this edge loss to encourage the prediction close to a sparse edge image. However, we also do not want the predicted sketch image containing inconsistent edges. Hence, we can use a pre-trained VGG19 [25] model (gray box in Figure 5) by keeping all convolutional layers and discarding the fully connected layers. To measure the style similarity, we can use VGG-19 to extract feature maps (*relu1_2*, *relu2_2*, *relu3_4*, *relu4_1*) for the input photo **X** and the predicted sketch **Y** as $V_{\mathbf{Y}}$ and $V_{\mathbf{X}}$. We firstly measure the **content loss** as

$$L_{\text{cont}} = \sum_{i=1}^{4} \|V^i_{\mathbf{Y}} - V^i_{\mathbf{X}}\|^1 \tag{2}$$

It measures the similarity of the sketch and photo at different feature levels. Its objective is to constrain the geometrical similarity between the sketch and photo to ensure that the predicted sketch has consistent contours and edges close to the photo. Furthermore, we compute the **style loss** between the predicted sketch and the sketch reference as,

$$L_{\text{style}} = \sum_{i=1}^{4} \|mean\left(V^i_{\mathbf{Y}}\right) - mean\left(V^i_{\mathbf{X}}\right)\|^1 + \|var\left(V^i_{\mathbf{Y}}\right) - var\left(V^i_{\mathbf{X}}\right)\|^1 \tag{3}$$

where mean(.) and var(.) are the mean and variance of the feature maps. Hence, this is to measure their mean and variance to align the sketch features close to the reference features.

Meanwhile, we have the discriminator to train together with the Photo2Sketch and Sketch2Photo. We have the **adversarial loss** L_{adv} to measure the perceptual similarity between the predicted sketch and the reference image. We have the Sketch2Photo to reconstruct the photo from the predicted sketch. We use l_1 errors to measure the **pixel loss** as L_{pixel}. Since we have identical encoder-decoder structure for Photo2Sketch and Sketch2Photo, we can also use the **feature loss** to further constrain the feature similarity between the predicted photo **X'** and input photo **X** as,

$$L_{\text{feat}} = \|W_{\mathbf{X}'} - W_{\mathbf{X}}\|^1 \tag{4}$$

where $W_{\mathbf{X}'}$ and $W_{\mathbf{X}}$ are the features of predicted photo and input photo from the encoder.

Finally, we have the total loss combining all the losses as,

$$\begin{aligned} L =& \lambda_{edge} L_{\text{edge}} + \lambda_{cont} L_{\text{cont}} + \lambda_{style} L_{\text{style}} + \\ & \lambda_{adv} L_{\text{adv}} + \lambda_{pixel} L_{\text{pixel}} + \lambda_{feat} L_{\text{feat}} \end{aligned} \tag{5}$$

where we have $\lambda_{(.)}$ as the weighting parameters to balance all the loss terms.

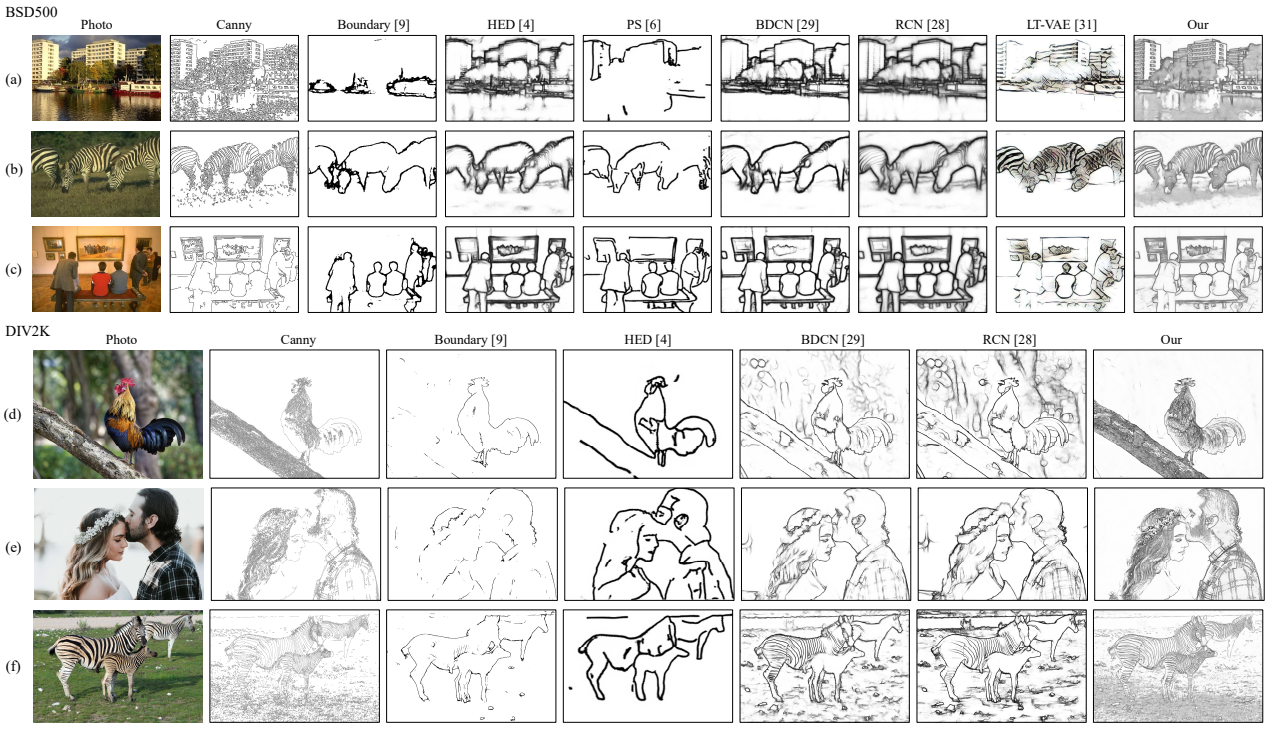

Fig. 6: **Visual comparison among different methods.** We use 3 examples from BSD500 and 3 examples from DIV2K for visualization. It can be seen that our method not only preserves the details on textures, it also generates different colors of lines that reflects the confidence of drawings.

IV. Experiments

A. Datasets and Settings

To train our network, we used images from MS-COCO [35] as training data and the images from ImageNet Sketch [34] as sketch references. At training, we keep the image ratio and crop a region of 256×256 as training patches. As data augmentation, we randomly flip images horizontally and vertically. To obtain the edge map for training data, we use the Canny detector in OpenCV package[1] to extract them. For the testing data, we consider two different sets of data: 1) edge based dataset with annotated edge maps and 2) large dataset with rich contents. For the former, we use BSD500 [6] dataset to estimate the performance of edge detection. BSD500 contains 500 images with medium resolutions, each image has one edge map annotated by humans. For the latter, we used DIV2K [28] dataset to estimate the perceptual quality of sketch images. DIV2K contains 800 images with 2K resolutions. The contents vary a lot. This is good, since it can be used to test the generalization of different approaches.

Implementation details. We trained our network in two stages: 1) we firstly trained the whole network without using the discriminator, and 2) then we fixed the Sketch2Photo and added the discriminator to train the whole network. Each stage we trained it for 100 epochs. We trained the network using Adam optimizer with a learning rate of 2×10^{-4} and a batch size of 8 for 12 hours on a single NVIDIA RTX2080 GPU using PyTorch.

B. Evaluation Metric

To evaluate our approach, we consider two criteria: boundary detection and perception similarity. For the former, we followed the standard protocol used in edge detection [4]-[19] to match predicted pixels to the ground truth, that is, **F1-score** used in [6, 26]. It is used to check the continuity of the boundaries.

Furthermore, for perception similarity, we assume that a good sketch image should contain key features that be recognized by a pre-trained deep feature extractor. We have three evaluation approaches, including **Laplacian loss**, **LPIPS** and **style loss**. The Laplacian loss [20] is to check the structural similarity between prediction and ground truth. It is computed on the Difference of Gaussian (DoG) to form pyramid edge maps. We used it to check the edge alignment of the sketch images. We also used LPIPS [33] to check the deep feature similarity between prediction and ground truth. The style loss is computed from the pre-trained VGG16 to measure feature correlations between prediction and ground truth[2]. Hence, we used MTCNN [32] to train a TV comedy show[3] for

[1]https://opencv.org/

[2]Following [33], we use pre-trained AlexNet to extract features to compute L1 distance between prediction and ground truth. The style loss was computed using Equation (3)

[3]We collected two episodes of Friends from Youtube and extract total 10000 frames for testing

TABLE I: **Comparison with state-of-the-art methods**. We compare our proposed method to other state-of-the-art methods on BSD500 testing dataset. Two sets of evaluations were used, including edge like criterion (F1, Pre and Rec) (the larger the better) and perception criterion (Lap, LPIPS and style loss) (the smaller the better)

Method	Edge eval.			Perception eval.		
	F1 ↑	Pre ↑	Rec ↑	Lap ↓	LPIPS ↓	style loss ↓
Canny	0.611	0.602	0.620	0.167	0.574	3.004
HED [4]	0.782	0.779	0.785	0.113	0.475	2.887
Pix2pix [6]	0.536	0.625	0.469	0.165	0.571	2.972
PS [6]	**0.822**	0.879	0.773	0.147	0.552	2.901
RCN [28]	0.819	**0.882**	0.764	0.102	0.469	2.865
BDCN [29]	0.806	0.821	**0.792**	0.116	0.472	2.798
LT-VAE [31]	0.602	0.592	0.612	0.201	0.605	2.799
Ours	0.807	0.833	0.783	**0.0803**	**0.465**	**2.034**

face detection. Then we used different approaches to generate different facial sketches and used them for face detection. A good sketch model should preserve the facial features that can be recognized by MTCNN. We evaluated the face detection by the number of missing detections and correct detections.

C. Overall Comparison

To compare the performance of our proposed method, let us use several state-of-the-art edge-like detection approaches: Canny detection (Canny) [1], Holistically-Nested Edge Detection (HED) [4], Pix2pixe [36], Photo-Skething (PS) [6], Richer Convolutional Features for Edge Detection (RCN) [28] and Bi-Directional Cascaded Network (BDCN) [29]. We also include one style transfer approach for comparison, LT-VAE [31][4]. We have the quantitative results in Table I.

In Table I, we compare different methods on BSD500 testing dataset with two different sets of evaluations: Edge evaluation and Perception evaluation. For edge evaluation, we followed [6, 26] to compute the Precision (Pre), Recall (Rec) and F1 score (F1)[5]. The Perception evaluation includes Laplacian loss (Lap), LPIPS and style loss. Note two observations from Table I:

1) For edge like evaluation, our method shows comparable results to other methods. However, note that HED, Pix2pix, PS, RCN and BDCN are designed specifically for edge detection and they were trained on dataset with annotated edge maps. It is obvious that they can achieve good results on F1 score, Precision and Recall. On the other hand, our method is to train in an unsupervised manner that no annotated edge maps are provided.

[4]We make use of LT-VAE for comparison because style transfer is a popular technology that can transfer the style appearance to the content image. Based on the idea of style transfer, we use one sketch reference as the style image and the testing photo as the content image, then input them to the pre-trained model to obtain sketch-like images. LT-VAE transfers styles only so it focuses more on appearance rather than edge generation, which is very different from our work.

[5]We use the ground truth edge map and the predicted sketch map to compute the precision and recall. The F1 score is computed as 2×(Precision×Recall)/(Precision+Recall)

TABLE II: **Comparison on Face detection using different methods**. We compare our proposed method to other state-of-the-art methods on face detection. We measure the performance by counting the number of miss detection and correct detection, then we compute the average accuracy.

Method	Canny	HED [4]	PS [6]	RCN [28]	BDCN [29]	LT-VAE [31]	Ours
Avg. missing ↓	3.40	3.51	3.75	3.21	2.89	2.45	**1.02**
Avg. correct ↑	0.76	0.43	0.42	0.62	0.67	0.72	**1.84**
Accuracy ↑	21.3%	20.4%	19.45%	21.8%	22.2%	26.2%	**77.2%**

2) For perception evaluation, our method shows superior performance compared to others, which indicates that our method preserves more content information. For example, our results on Laplacian loss (Lap) and LPIPS demonstrate that our method can align well with the ground truth edge map, while our result on style loss shows that our method can generate images close to sketch references in terms of perceptual quality. We also conducted an ablation study to test the effect of the loss terms in Equation (5). The results show that using edge loss encourages the prediction with rich edges, using content and style losses ensure that the prediction with similar sketch appearance, and using adversarial loss can fill or compensate some key features. Using feature loss can further constrain the prediction close to the input.

In order to further demonstrate the performance of different methods, we visualize a few examples from BSD500 and DIV2K datasets in Figure 6. We show three examples from BSD500 dataset and another three examples from DIV2K dataset. It can be seen that Canny does not clearly highlight the contour of the objects which makes the image messy. Boundary [9] and HED[4], on the other hand, only roughly depict the boundaries of the objects which do not describe fine details of the photos. BDCN [29] and RCN [28] can improve the edge detection with more details but they cannot perform consistently. For example, they do not show the strides on all zebras (Figure 6 f). They also cannot distinguish the 3D correlations in Figure 6 d. It is clear that this photo is a close-up shot that the key feature is a rooster. BDCN and RCN still detect the tree behind the rooster which makes the results less attractive.

D. Perceptual Comparison

Note again that our method is much different from other edge like approaches since our method focuses more on automate sketching rather than just detecting the edges. It is important to keep key features as much as possible. However, it is rather subjective to measure the goodness of a sketch. To further demonstrate the superior performance of our method, we tested different methods on two episodes of TV shows and used MTCNN [32] to detect faces. A good sketch generation model should preserve the facial features that can be recognized by the face detector. We measure the face detection ability by counting the number of missing detections and correct detections.

From Table II, we can find that our proposed method outperforms others by a large margin because our method can preserve the facial features that can be recognized by the

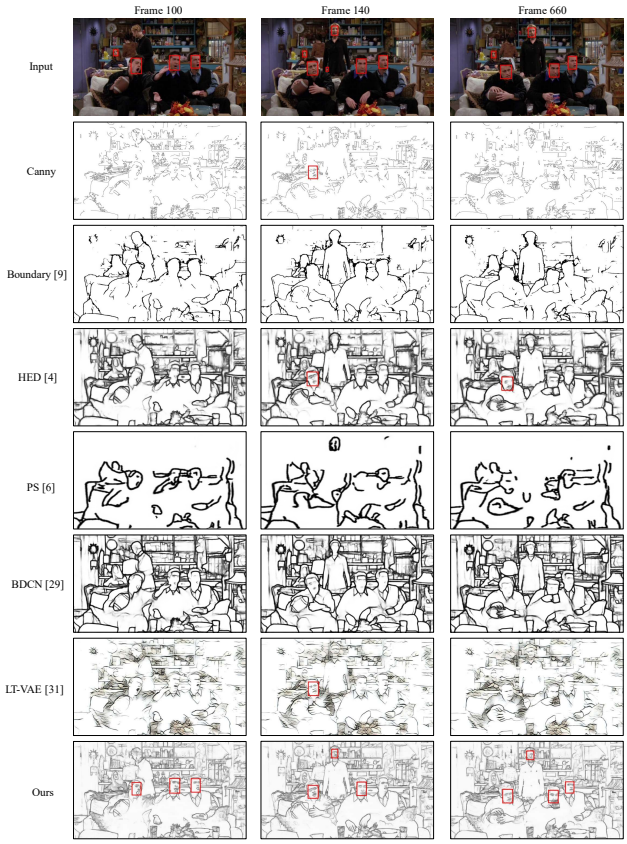

Fig. 7: **Comparison of face detection using different methods.** We visualize 3 frames from the TV show for face detection. We marked the detected face in red boxes. It can be seen that using our approach for sketch generation does not affect much for face detection.

face detector. Let us visualize some face detection results in Figure 7. It is worth noticing that using MTCNN can still detect the faces from our sketch results while other methods fail because they cannot preserve the facial features. The reason is that other methods only focus on strong edges of furniture or other objects and overlook the fine facial feature.

E. Multiple sketch generation

As described in Equation (1), our proposed method can generate C sketch candidates for sampling. All sketch candidates are weighted to form the final output. The weighting values are learned by the Softmax function so that the sum of the weights is equal to 1. In our design, users can randomly select one of the candidates for evaluation. In our experiments, we set C=5 so users can generate 5 different sketch results for preference. We demonstrate different sketch results in Figure 8. Two samples from DIV2K are used to visualize the sketch results. It can be found that different sketches reveal different degrees of details and grayscale distributions.

V. Conclusions

We propose a uniform and robust model to transfer photos to sketches. There are three major modules in our methods: 1) we form a close loop between photos and sketches so the model can supervise itself to generate photo-realistic sketch image, 2) we introduce sketch references to encourage the model to generate results close to the references, and 3) we use a soft weighted layer to generate different candidates so users can have multiple choices. Unlike most existing edge like approaches, our proposed approach can cast real photos into sketch-like images while preserving more visual details and key features. Though our proposed method is trained in an unsupervised manner, it still outperforms most edge detection algorithms. Possible future work includes video based sketch generation and interactive sketch manipulation.

Acknowledgment

This work was supported by Caritas Institute of Higher Education, The Hong Kong Polytechnic University, and Research Grants Council of the Hong Kong Special Administrative Region, China (under Grant Nos. UGC/IDS(R)11/19 for computing power support and UGC/IDS(C)11/E0/120 for manpower support.)

References

[1] Canny, J., A Computational Approach To Edge Detection, IEEE Transactions on Pattern Analysis and Machine Intelligence, 8(6):679–698, 1986.
[2] N. Kanopoulos, N. Vasanthavada and R. L. Baker, "Design of an image edge detection filter using the Sobel operator," in IEEE Journal of Solid-State Circuits, vol. 23, no. 2, pp. 358-367, April 1988, doi: 10.1109/4.996.
[3] Y. Liu and M. S. Lew, "Learning Relaxed Deep Supervision for Better Edge Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 231-240.
[4] S. Xie and Z. Tu, "Holistically-Nested Edge Detection," 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1395-1403, doi: 10.1109/ICCV.2015.164.
[5] Y. Wang, X. Zhao and K. Huang, "Deep Crisp Boundaries," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1724-1732, doi: 10.1109/CVPR.2017.187.
[6] M. Li, Z. Lin, R. Mech, E. Yumer and D. Ramanan, "Photo-Sketching: Inferring Contour Drawings From Images," 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), 2019, pp. 1403-1412, doi: 10.1109/WACV.2019.00154.
[7] S. Gupta, P. Arbeláez and J. Malik, "Perceptual Organization and Recognition of Indoor Scenes from RGB-D Images," 2013 IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 564-571, doi: 10.1109/CVPR.2013.79.
[8] Wei Shen, Xinggang Wang, Yan Wang, Xiang Bai and Z. Zhang, "DeepContour: A deep convolutional feature learned by positive-sharing loss for contour detection," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3982-3991, doi: 10.1109/CVPR.2015.7299024.
[9] J. Yang, B. Price, S. Cohen, H. Lee and M. Yang, "Object Contour Detection with a Fully Convolutional Encoder-Decoder Network," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 193-202, doi: 10.1109/CVPR.2016.28.
[10] A. Khoreva, R. Benenson, M. Omran, M. Hein and B. Schiele, "Weakly Supervised Object Boundaries," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 183-192, doi: 10.1109/CVPR.2016.27.
[11] D. Acuna, A. Kar and S. Fidler, "Devil Is in the Edges: Learning Semantic Boundaries From Noisy Annotations," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 11067-11075, doi: 10.1109/CVPR.2019.01133.
[12] Hu Z., Zhen M., Bai X., Fu H., Tai C. (2020) JSENet: Joint Semantic Segmentation and Edge Detection Network for 3D Point Clouds. In: Vedaldi A., Bischof H., Brox T., Frahm JM. (eds) Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science, vol 12365. Springer, Cham.

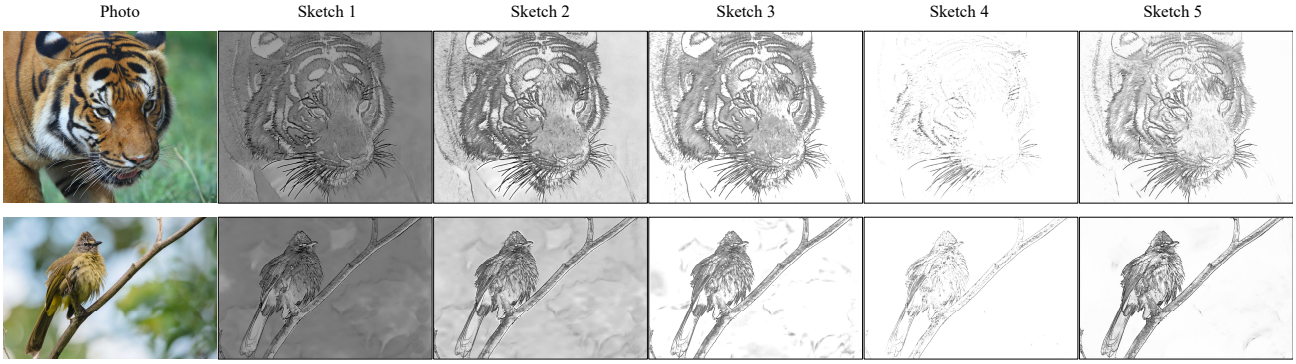

Fig. 8: **Different sketch samples generated by our proposed method.** We visualize 5 samples of sketch images generated by our method. Different sketch images depict different contents.

[13] David Ha, Douglas Eck, "A Neural Representation of Sketch Drawings," International Conference on Learning Representations, 2018.

[14] Cao, N., Yan, X., Shi, Y., and Chen, C. AI-Sketcher : A Deep Generative Model for Producing High-Quality Sketches. Proceedings of the AAAI Conference on Artificial Intelligence, 33(01), 2564-2571, 2019.

[15] Ayan Kumar Bhunia, Ayan Das, Umar Riaz Muhammad, Yongxin Yang, Timothy M. Hospedales, Tao Xiang, Yulia Gryaditskaya, and Yi-Zhe Song. 2020. Pixelor: a competitive sketching AI agent. so you think you can sketch? ACM Trans. Graph. 39, 6, Article 166 (December 2020), 15 pages.

[16] Songwei Ge and Vedanuj Goswami and C. Lawrence Zitnick and Devi Parikh, "Creative Sketch Generation," International Conference on Learning Representations, 2021.

[17] J. Song, K. Pang, Y. Song, T. Xiang and T. M. Hospedales, "Learning to Sketch with Shortcut Cycle Consistency," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 801-810, doi: 10.1109/CVPR.2018.00090.

[18] Wang, T., Toh, W.Q., Zhang, H., Sui, X., Li, S., Liu, Y., and Jing, W. RoboCoDraw: Robotic Avatar Drawing with GAN-based Style Transfer and Time-efficient Path Optimization. ArXiv, abs/1912.05099, 2020.

[19] Tong, Z., Chen, X., Ni, B., and Wang, X. Sketch Generation with Drawing Process Guided by Vector Flow and Grayscale. Proceedings of the AAAI Conference on Artificial Intelligence, 35(1), 609-616, 2021.

[20] Zhi-Song Liu, Li-Wen Wang, Chu-Tak Li, Wan-Chi Siu and Yui-Lam Chan, "Image Super-Resolution via Attention Based Back Projection Networks," 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), 2019, pp. 3517-3525, doi: 10.1109/ICCVW.2019.00436.

[21] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems (p./pp. 5998–6008).

[22] X. Wang, R. Girshick, A. Gupta and K. He, "Non-local Neural Networks," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 7794-7803, doi: 10.1109/CVPR.2018.00813.

[23] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks", in IEEE International Conference on Computer Vision (ICCV), 2017.

[24] Karen Simonyan and Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," International Conference on Learning Representations, 2015.

[25] Timofte, Radu and Gu, Shuhang and Wu, Jiqing and Van Gool, Luc and Zhang, Lei and Yang, Ming-Hsuan and Haris, Muhammad and others, "NTIRE 2018 Challenge on Single Image Super-Resolution: Methods and Results," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, June, 2018.

[26] X. Hou, A. L. Yuille, and C. Koch. "Boundary detection benchmarking: Beyond f-measures," The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2123–2130, 2013.

[27] D. Martin and C. Fowlkes and D. Tal and J. Malik, "A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics," Proceedings IEEE International Conference Computer Vision, vol. 2, pp. 416–423, July, 2001.

[28] Y. Liu et al., "Richer Convolutional Features for Edge Detection," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 41, no. 8, pp. 1939-1946, 1 Aug. 2019, doi: 10.1109/TPAMI.2018.2878849.

[29] J. He, S. Zhang, M. Yang, Y. Shan and T. Huang, "BDCN: Bi-Directional Cascade Network for Perceptual Edge Detection," in IEEE Transactions on Pattern Analysis and Machine Intelligence, doi: 10.1109/TPAMI.2020.3007074.

[30] Huang, Xun and Belongie, Serge, "Arbitrary style transfer in real-time with adaptive instance normalization," Proceedings IEEE International Conference Computer Vision, 2017.

[31] Zhi-Song Liu, Vicky Kalogeiton and Marie-Paule Cani, "Multiple Style Transfer via Variational AutoEncoder," 2021 IEEE International Conference on Image Processing, Oct, 2021.

[32] F. Schroff, D. Kalenichenko and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 815-823, doi: 10.1109/CVPR.2015.7298682.

[33] R. Zhang, P. Isola, A. A. Efros, E. Shechtman and O. Wang, "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 586-595, doi: 10.1109/CVPR.2018.00068.

[34] Haohan Wang, Songwei Ge, Zachary C. Lipton, Eric P. Xing, "Learning Robust Global Representations by Penalizing Local Predictive Power," NeurIPS 2019: 10506-10518

[35] Lin TY. et al, "Microsoft COCO: Common Objects in Context," IEEE European Conference on Computer Vision, vol 8693, 2014.

[36] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. CVPR, 2017.

[37] Choy, S. S. O., Choy, S. T., Wan-Chi Siu. New single-pass algorithm for parallel thinning. Computer Vision and Image Understanding, 62(1), 69-77, 1995.

[38] Pui-Kin Ser and Wan-Chi Siu. Memory compression for straight line recognition using the Hough transform. Pattern Recognition Letter, 16, 2, pp. 133–145, Feb. 1995.

[39] Chun-pong Chau and Wan-Chi Siu, New Nonparametric Dominant point Detection, pp.363-374, vol. 148, no. 5, IEE Proceedings, Vision, Image and Signal processing, Oct 2001, UK.

[40] R. Morais, V. Le, T. Tran, B. Saha, M. Mansour and S. Venkatesh, "Learning Regularity in Skeleton Trajectories for Anomaly Detection in Videos," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 11988-11996.

[41] W. Shen, K. Zhao, Y. Jiang, Y. Wang, X. Bai and A. Yuille, "DeepSkeleton: Learning Multi-Task Scale-Associated Deep Side Outputs for Object Skeleton Extraction in Natural Images," in IEEE Transactions on Image Processing, vol. 26, no. 11, pp. 5298-5311, Nov. 2017.

[42] Y. Wang, Y. Xu, S. Tsogkas, X. Bai, S. Dickinson and K. Siddiqi, "DeepFlux for Skeletons in the Wild," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 5282-5291.

[43] C. Liu, Y. Tian, Z. Chen, J. Jiao and Q. Ye, "Adaptive Linear Span Network for Object Skeleton Detection," in IEEE Transactions on Image Processing, vol. 30, pp. 5096-5108, 2021.