Sketch Plus Colorization Deep Convolutional Neural Networks for Photos Generation from Sketches

Vinnia Kemala Putri[1] and Mohamad Ivan Fanany[2]
[1,2]Machine Learning and Computer Vision Laboratory
[1,2]Faculty of Computer Science, Universitas Indonesia
Depok, West-Java Indonesia
Corresponding Authors: vinnia.kemala51@ui.ac.id[1], ivan@cs.ui.ac.id[2]

Abstract—**In this paper, we introduce a method to generate photos from sketches using Deep Convolutional Neural Networks (DCNN). This research proposes a method by combining a network to invert sketches into photos (sketch inversion net) with a network to predict color given grayscale images (colorization net). By using this method, the quality of generated photos is expected to be more similar to the actual photos. We first artificially constructed uncontrolled conditions for the dataset. The dataset, which consists of hand-drawn sketches and their corresponding photos, were pre-processed using several data augmentation techniques to train the models in addressing the issues of rotation, scaling, shape, noise, and positioning. Validation was measured using two types of similarity measurements: pixel-difference based and human visual system (HVS) which mimics human perception in evaluating the quality of an image. The pixel-difference based metric consists of Mean Squared Error (MSE) and Peak Signal-to-Noise Ratio (PSNR) while the HVS consists of Universal Image Quality Index (UIQI) and Structural Similarity (SSIM). Our method gives the best quality of generated photos for all measures (844.04 for MSE, 19.06 for PSNR, 0.47 for UIQI, and 0.66 for SSIM).**

Keywords—sketch inversion; colorization; deep convolutional neural networks

I. INTRODUCTION

Face identification is one of the crucial issues, especially for law enforcement. Police department utilizes this technology to search for suspects on the run and missing people. Unfortunately, the photos of the suspects are not always available. The sketches of the suspects drawn by artists based on the information of eyewitnesses are used as substitutes of photos to recognize and identify suspects. However, the direct comparison of sketches and photos is difficult to do because of the significant difference between those images [1].

Prior works handled this issue by synthesizing photos into sketches and conducting sketch recognition. Wang and Tang [2] proposed a multiscale Markov Random Field model to synthesize sketches from photos. Wang et al., [3] used the patch-based and transductive method with a probabilistic model. By using pairs of photos-sketches and photos only as training data, they generated sketches from photos that did not have sketches. Nevertheless, methods that had been proposed still overlooked some important details that can affect the ability of the model to recognize faces using sketches.

Recent studies developed methods of sketches synthesis using deep learning to overcome the limitations of those conventional methods. Deep learning is the latest technology of machine learning that can extract features from low-level to high-level features. Zhang et al., [4] had used discriminative regularization DCNN for improving the ability of DCNN to distinguish the generated sketches from actual sketches. Simo-Serra et al., [5] used DCNN to simplify rough sketches.

But, the use of sketches for straightforward matching with photos from the database was not very effective because all those photos needed to be transformed into sketches before conducted sketch recognition. Moreover, sketches lack detailed information than photos, i.e. texture and color information. Hence, the need for transforming sketches into photos becomes mandatory.

Güçlütürk et al., [6] proposed a method to invert sketches into photos using DCNN as images generator and pre-trained model VGG-16 as a fixed features extractor [7]. They developed three models that were trained using large-scale faces in the wild dataset. Their model could synthesize line, shape, and texture well while the color was not synthesized accurately. Moreover, the models were trained using sketches which were generated by a simple image processing method, not the actual hand-drawn sketches.

In this paper, we propose the combination of a model to synthesize photos from sketches (sketch inversion) and a model to predict color in CIELab color space given grayscale images (colorization) DCNN to enhance the quality of color from generated photos so that it will be more similar to the actual photos. The sketch inversion net is an implementation of [6] while the colorization net is inspired by [8] and [9]. The use of CIELab color space for colorization tasks was proposed by [8] whereas the cascade architecture of convolutional neural networks (CNN) for colorization net is inspired by [9].

In addition to embedding colorization DCNN into sketch inversion DCNN, in this paper, we also present thorough analysis on the effects of data augmentation techniques [10] to the reconstruction performance. The augmentations are aimed to allow the models to learn from hand-drawn sketches and their corresponding photos in uncontrolled conditions. These conditions were characterized by changes in geometric and spatial conditions which were created using several data

978-1-5386-0549-3/17/$31.00 ©2017 IEEE

augmentation techniques to tackle the problem of positioning, scaling, and rotation.

The paper is structured as follows: Section II describes the explanations of data and its augmentation method used in this paper. Section III explains the architecture of the models and their use. Section IV consists of experiments and its results conducted in this research. Section V presents the conclusion of this research.

II. Dataset and Data Augmentation

A. Original Dataset

We made use of the CUHK dataset [2]. It contains 188 pairs of frontal face photos and their corresponding hand-drawn sketches from the Chinese University of Hong Kong database. One photo represents one identity. The size of each image was changed to 96 × 96 pixels. As for pre-preprocessing the image, we only subtracted per-channel mean over the training set from each channel. So, we trained our networks on the centered raw RGB values.

B. Data Augmentation

The easiest and most common method to overcome the lack of image data is to create new data by perturbing the original data artificially. Each image was perturbed before fed into the network by applying 10 data augmentation techniques to each image, i.e. scale, translation, rotation, shear, projective transformation, blur, flip, color jittering/noise, overlap crop and waving. These increased the size of our data into 17,108 images.

III. Models

A. Overall Architecture

Figure 1 summarizes the architecture of our system. Overall the system contained two networks such as sketch inversion net for inverting face sketches to synthesize face photos and colorization net for color prediction task of grayscale images. Sketch inversion net used sketches as input data and photos as targets. It generated RGB photos as prediction images. Meanwhile colorization net used grayscale images, which were transformed from generated photos, as input data and photos in CIELab as targets. This network delivered the prediction of color in CIELab color space. The details of input, prediction and target images of each network are shown in Figure 2.

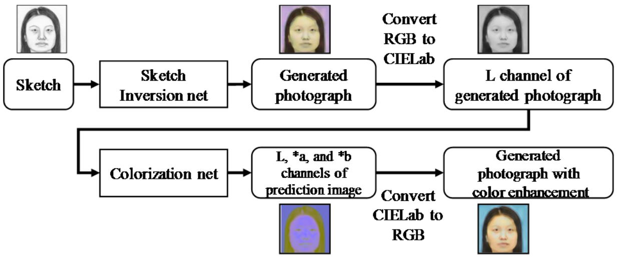
Sketch
Sketch Inversion net
Generated photograph
Convert RGB to CIELab
L channel of generated photograph
Colorization net
L, *a, and *b channels of prediction image
Convert CIELab to RGB
Generated photograph with color enhancement

Fig. 1. System diagram

To combine sketch inversion net and colorization net, firstly we needed to generalize the color space of each network. Because the output of sketch inversion net is a generated photo in RGB color space, it is necessary to convert it into CIELab color space. So, the generated photo would have L, *a, and *b channels. Next, we pull the L channel out and then presented it as an input image to the colorization net. The network would predict L, *a, and *b channels of the input image then generated a photo in CIELab color space. We used CILEab color space because the L channel of CIELab is already in grayscale, as shown in left image of Figure 2b, so the model only need to predict the *a and *b channels of the output images since the L channel of output images is the same as the one for input images. The last step was to convert its color space back, from CIELab to RGB. Therefore, we would get a generated photo from the sketch with color enhancement.

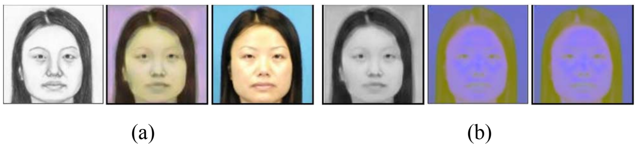

Fig. 2. Input, prediction, and target images for: (a) sketch inversion net and (b) colorization net.

B. Sketch Inversion Net

For sketch inversion net, we implemented the method proposed by [6]. The method is summarized in Figure 3.

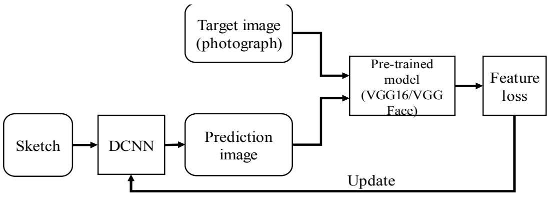

Fig. 3. Sketch inversion net

It comprised two types of CNN which were pre-trained model 16-layers VGG network (VGG-16/VGGFace [11]) and DCNN proposed by [6]. We made use of VGG network as a fixed features extractor and to calculate loss between prediction images and target images. Meanwhile, the DCNN served as a generator to synthesize photos from sketches. The architecture of DCNN was based on [6].

The generated prediction images and photos as target images were fed-forward into VGG network sequentially. The outputs of second convolution layer after first pooling layer of VGG network (conv2_2) were used for representing the features of the targets and predictions. The use of conv2_2 outputs as a features representation was based on [12] for reconstructing input images from different output layers. The visualization of reconstructed images became more like blobby things than the actual input images as the output layer went deeper. Thus, we chose a layer that was not too deep and not too shallow so that it could extract features which had a good representation and at the same time could give a good visualization of the reconstructed images.

To generate a prediction image that matched the target image, we minimized the error between feature representation of prediction and the target. So, let $\phi(t)_{i,j,k}$ and $\phi(y)_{i,j,k}$ be the feature representation of the target and the prediction consecutively, and n is the total number of features. We defined the feature loss ℓ_f as the mean of sum-squared-error between two feature representation, as in

$$\ell_f = \frac{1}{n}\sum_{i,j,k}(\phi(t)_{i,j,k} - \phi(y)_{i,j,k})^2 \quad (1)$$

where i and j are height and width of the feature maps respectively, and k is the total number of feature maps channels. The feature loss was then back-propagated through DCNN and was used by DCNN to learn how to generate an image which had feature representation that matched the feature representation of the target image. The feed-forward & back-propagation processes were repeated until the error reached the minimum.

C. Colorization Net

The colorization net was used to correct and enhance the color of the generated photos from sketch inversion net. It contained four CNNs arranged in cascade as shown in Figure 4. A CNN with a very deep architecture is much more difficult to train than the CNN with a simple and shallow architecture. Unfortunately, the shallow architecture cannot extract features representation as good as the one with deep architecture. Several CNNs with simple architectures are arranged in cascade to resolve this issue. The main idea is the next CNN should get a boost and can generate an image based on the previous CNN. Furthermore, the training of each CNN can be easier because of the simple and shallow architecture of each CNN. The architecture for each CNN was based on [9].

Each of CNNs served as a generator. The first and last CNNs were convolution generators and the middle two were deconvolution generators. The grayscale of generated photo was fed forward into first CNN then the CNN generated a photo with its color prediction in RGB. We minimized the error between prediction image and target image to get a prediction image that matched target closely. So, let $(t)_{n,m,k}$ be the target image and $(y)_{n,m,k}$ is prediction image. The pixel loss ℓ_p between target and prediction is defined as follow

$$\ell_p = \frac{1}{P}\frac{1}{N}\frac{1}{M}\sum_{p=0}^{P-1}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}((t)_{m,n,p} - (y)_{m,n,p})^2 \quad (2)$$

where M and N are height and width of the image respectively, and P is the total number of channels. The pixel loss was then back-propagated through the CNN and was used to learn how to generate an image which had color representation that closely matched the color representation of the target image. The feed-forward & back-propagation processes were repeated until the error reached the minimum.

For next CNN, the input image was the prediction image from previous CNN. Next CNN also used pixel loss from previous CNN. For example, the total pixel loss for CNN 2 consisted of pixel loss of CNN 1 (pixel loss I) and pixel loss of CNN 2 (pixel loss II). The processes of feed-forward & backpropagation for next CNN is the same as the previous one.

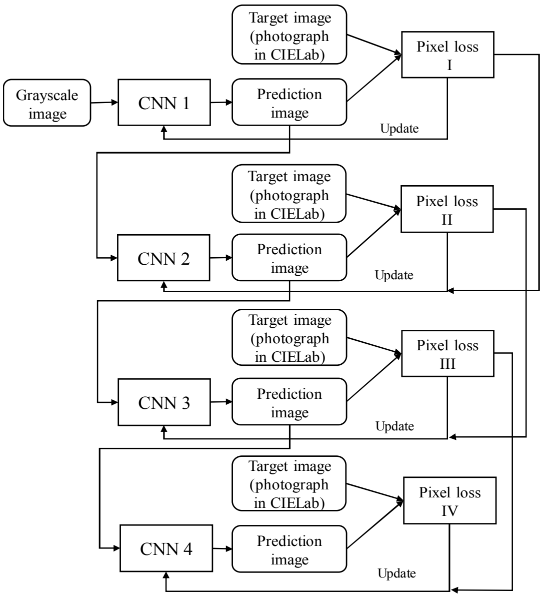

Fig. 4. The proposed colorization net

D. Validation

We compared the generated photos with ground truth photos to find out whether the generated photos matched ground truth closely or not. We also compared the performance of our model with [6] by comparing the quality of photos that were generated using our model and model proposed by [6]. We quantitatively measured the quality of the generated photos using two types of similarity measurements such as pixel-difference based, i.e. MSE and PSNR, and HVS algorithm which mimics human perception in evaluating the quality of an image, i.e. UIQI and SSIM.

MSE [13] is defined as the average of sum-squared error between prediction and target. The lower the MSE is, the better it is. Its minimum value is 0 while the maximum is square of the range between minimum and maximum values of a pixel, e.g. for an 8-bit image the maximum MSE is 255^2. MSE is defined as in

$$MSE = \frac{1}{P}\frac{1}{N}\frac{1}{M}\sum_{p=0}^{P-1}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}((t)_{m,n,p} - (y)_{m,n,p})^2 \quad (4)$$

PSNR [14] measures the composition of pixels between the prediction image and the target. It defines prediction as a signal and its difference values from the target as noise. The higher the PSNR is, the better it is. Its minimum value is 0 while the maximum is infinity when the prediction is perfectly similar to the target. The definition of PSNR is expressed as in

$$PSNR = 10log\frac{s^2}{MSE} \quad (5)$$

where s = 255 for an 8-bit image.

UIQI [15] uses three factors to evaluate the quality of an image, i.e. luminance, contrast, and structure:

$$UIQI(x,y) = \frac{4\mu_x\mu_y\sigma_{xy}}{(\mu_x^2+\mu_y^2)(\sigma_x^2+\sigma_y^2)} \quad (6)$$

where μ_x, μ_y, σ_x, σ_y, and σ_{xy} are means, standard deviations, and covariance of target and prediction. The higher the UIQI is, the better it is. When the prediction is perfectly similar to the target, its value is 1.0. Its value is 0.0 when the prediction is completely different from the target.

SSIM [16] is patch-based similarity measurement that measures perceptual quality of an image regarding contrast, luminance, and structure. The definition of SSIM for each patch as follows

$$SSIM(x,y) = \frac{(2\mu_x\mu_y+c_1)(2\sigma_{xy}+c_2)}{(\mu_x^2+\mu_y^2+c_1)(\sigma_x^2+\sigma_y^2+c_2)} \quad (7)$$

where μ_x, μ_y, σ_x, σ_y, and σ_{xy} are means, standard deviations, and covariance of target and prediction. Furthermore, $c_1 = (0{,}01\ s)^2$ and $c_2 = (0{,}03\ s)^2$ with $s = 255$ for an 8-bit image. Let p as a total number of patches, the SSIM for an image is defined as the mean of sum of the SSIMs for patches:

$$MSSIM = \frac{1}{p}\sum_{j=1}^{P} SSIM_j \quad (8)$$

The higher the SSIM is, the better it is. When the prediction is perfectly similar to the target, its value is 1.0. Its value is 0.0 when the prediction is completely different from the target.

IV. Experiments and Results

The models were trained on a single NVIDIA GeForce GTX Titan X Maxwell GPU 12GB GDDR5. The experiments comprise four scenarios. The details of each experiment are as follows.

A. Determine the Necessity of Data Augmentation

This experiment aimed to discover the necessity to augment the data and its effect in the training process. We trained two models. The model I was trained by original dataset while the model II was trained by the augmented dataset. Each model used 100 data as the training set. Both of models were conducted in two testing phases. The first phase, both of models were tested using 88 original data. For the second phase, both of models were tested using 5,133 augmented data. The results of the experiment are shown in Table I and Table II respectively.

TABLE I. Comparisons of Similarity Between Generated Images and Ground Truth for Testing Phase I

Model	Ave-MSE	Ave-PSNR	Ave-UIQI	Ave-SSIM
Model I	**1,040.78**	**18.11**	**0.45**	**0.66**
Model II	1,151.16	17.65	0.44	0.64

Based on Table I, the model I generated better photos than model II when the condition of the testing set was not perturbed. But when the geometric and spatial conditions of the testing set were perturbed as shown in Table II, the model I tended to fail in synthesizing photos from sketches while model II could synthesize well. It showed that the model II which was trained using augmented data was more robust than model I which was trained without augmented data.

TABLE II. Comparisons of Similarity Between Generated Images and Ground Truth for Testing Phase II

Model	Ave-MSE	Ave-PSNR	Ave-UIQI	Ave-SSIM
Model I	8,063.52	9.67	0.07	0.20
Model II	**1,595.05**	**17.20**	**0.41**	**0.56**

B. Experiment on Finding the Most Significance Data Augmentation Technique

The experiment was conducted to find which data augmentation techniques used in the training process, that gave the most considerable influence on the outputs of models. We trained ten models, and each model was trained by one of ten techniques and the other nine techniques served as a testing set. Each technique comprised four variations, so each model had 752 data as training set and 6.768 data as a testing set. Based on Table III, the technique that gave the best result was translation (1,216.67 for MSE, 17.83 for PSNR, 0.46 for UIQI, and 0.62 for SSIM).

TABLE III. Comparisons of Data Augmentation Techniques to the Quality of Generated Images

Data augmentation techniques	Ave-MSE	Ave-PSNR	Ave-UIQI	Ave-SSIM
Blur	1,611.69	16.37	0.42	0.56
Color jittering/Noise	1,596.65	16.40	0.41	0.54
Flip	1,460.83	16.69	0.41	0.55
Overlap crop	1,614.84	16.27	0.39	0.53
Projective transform	1,399.20	17.02	0.41	0.58
Rotation	**1,202.07**	17.63	0.43	0.58
Shear	1,257.92	17.52	0.43	0.60
Translation	1,216.67	**17.83**	**0.46**	**0.62**
Wave	1,271.15	17.43	0.42	0.58
Zoom	1,357.46	17.18	0.43	0.58

C. Comparison of Previous Method and Ground Truth

The experiment was carried to train sketch inversion net by implementing method proposed by [6] and compared the generated photos with ground truth. In this experiment, we trained two models. Each model used different pre-trained model. The model I used VGGFace pre-trained model while Model II used VGG-16 pre-trained model. Both of models were trained using augmented data. A total number of data are 17,108 data with 70% was used for training (11,975 data) and 30% was used for testing (5,133 data).

Figure 5 visualizes some generated photos from both models. The first column visualizes sketches as input data. The second column is generated photos from Model I that used VGGFace pre-trained model. The third column is generated photos from Model II that used VGG-16 pre-trained model. The fourth column is ground truth photos. As shown in Figure 5, the use of sketch inversion net without colorization net could synthesize other elements e.g. line, shape, form, and texture well while the color was not synthesized accurately. The model that used VGGFace pre-trained model could not synthesize color as good as the model that used VGG-16 pre-trained model.

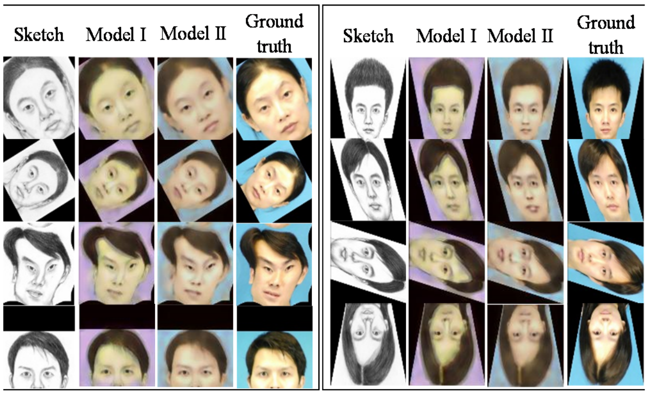

Fig. 5. Comparison of generated photos between two models

Based on Table IV, the first model gave a performance as good as the second model regarding HVS measurement (UIQI and SSIM). But regarding pixel difference-based measurement, the first model suffered a setback because it could not synthesize color well which led to the increase of error values of each pixel. By combining sketch inversion net and colorization net, the issue of color synthesizing error could be tackled, and the performance of the first model could be improved.

TABLE IV. COMPARISONS OF SIMILARITY BETWEEN GENERATED PHOTOS AND GROUND TRUTH

Model	Ave-MSE	Ave-PSNR	Ave-UIQI	Ave-SSIM
Model I (pre-trained model: VGGFace)	1,182.08	17.54	**0.46**	**0.65**
Model II (pre-trained model VGG-16)	**1,047.45**	**18.09**	**0.46**	**0.65**

D. Comparison with the State of the Art

This experiment was carried to implement the proposed method then compared it with [6]. We trained colorization net then combined it with sketch inversion net that was trained previously. Based on the evaluation of the previous experiment, colorization net was trained using generated photos from sketch inversion net that used VGG-16 pre-trained model. A total number of data are 17,108 data with 70% as the training set (11,975 data) and 30% as the testing set (5,133 data).

As shown in Figure 4, every CNN in colorization net generates prediction images. Table V presents the comparison between ground truth photos and generated photos and for each CNN. Figure 6 and 7 shows comparisons of generated photos for each CNN in colorization net using input images from sketch inversion net with VGG-16 and VGGFace pre-trained model respectively. The best model was achieved by a combination of sketch inversion net using VGGFace and colorization net using outputs of CNN 4.

TABLE V. COMPARISONS OF PROPOSED MODELS

Model	Ave-MSE	Ave-PSNR	Ave-UIQI	Ave-SSIM
Colorization net – output CNN 1				
Sketch Inv.(VGGFace) + Colorization	1,205.31	17.46	0.45	0.60
Sketch Inv.(VGG-16) + Colorization	1,091.85	17.85	0.46	0.60
Colorization net – output CNN 2				
Sketch Inv.(VGGFace) + Colorization	859.31	18.97	0.47	0.65
Sketch Inv.(VGG-16) + Colorization	857.82	18.97	0.48	0.65
Colorization net – output CNN 3				
Sketch Inv.(VGGFace) + Colorization	855.76	18.99	0.47	0.66
Sketch Inv.(VGG-16) + Colorization	857.12	18.98	0.48	0.65
Colorization net – output CNN 4				
Sketch Inv.(VGGFace) + Colorization	**844.04**	**19.06**	0.47	**0.66**
Sketch Inv.(VGG-16) + Colorization	848.75	19.03	**0.48**	0.65

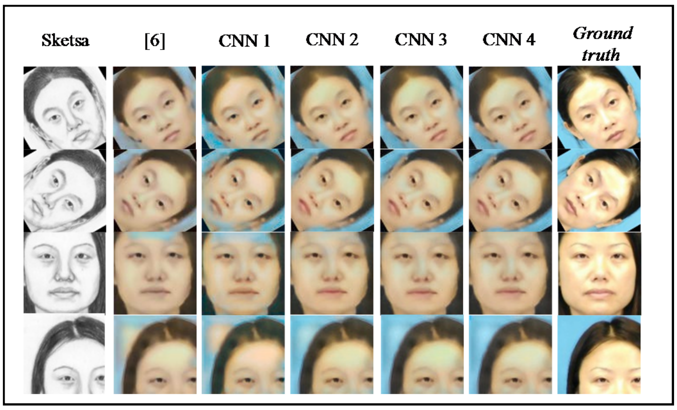

Fig. 6. Comparison of generated photos for each CNN using input images from sketch inversion with VGG-16 pre-trained model

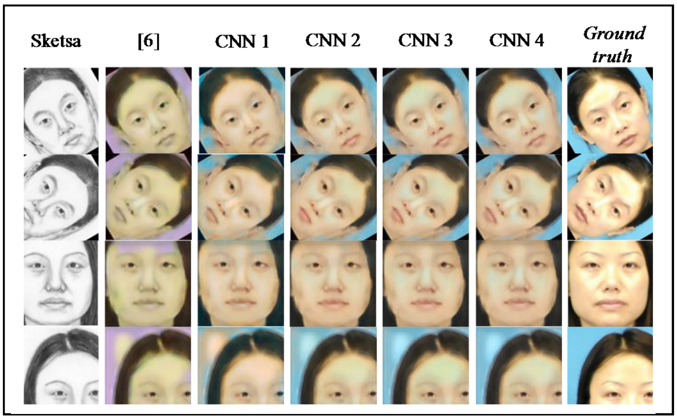

Fig. 7. Comparison of generated photos for each CNN using input images from sketch inversion with VGGFace pre-trained model

Table VI shows the comparison of the performance of proposed model and [6]. Compared with [6] using VGGFace pre-trained model, our proposed method not only increased PSNR by 1.52, UIQI by 0.01, and SSIM by 0.01 but also decreased MSE by 338.04. When compared with [6] using VGG-16, our proposed method increased PSNR by 0.97, UIQI by 0.01, and SSIM by 0.01. It correspondingly decreased MSE by 203.41.

TABLE VI. COMPARISONS OF PROPOSED METHOD AND METHOD PROPOSED BY [6]

Model	Ave-MSE	Ave-PSNR	Ave-UIQI	Ave-SSIM
[6] (pre-trained model: VGGFace)	1,182.08	17.54	0.46	0.65
[6] (pre-trained model VGG-16)	1,047.45	18.09	0.46	0.65
Proposed method	**844.04**	**19.06**	**0.47**	**0.66**

We found that by combining sketch inversion net and colorization net, it could solve photos generation problem for the overlap-crop type of data (Figure 8). We observed that the forehead area that was not accompanied by dark-edged or hair strokes on the upper side of the face was not generated well (Figure 8, second column). Our method could tackle that problem as shown in the third column of Figure 8 (a) and (b).

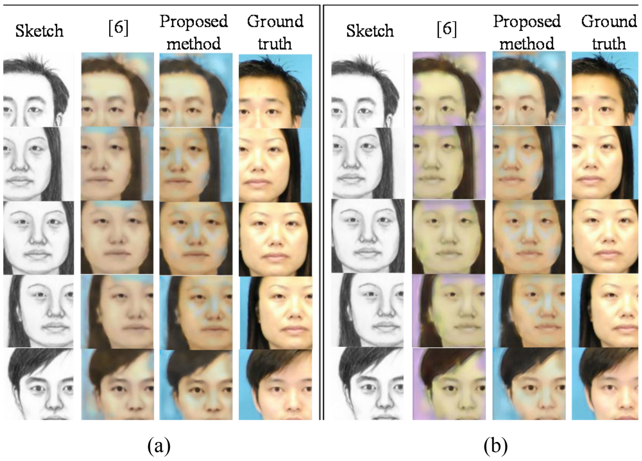

Fig. 8. Comparison of generated photos between proposed method and [6] using: (a) VGG-16 and (b) VGGFace pre-trained models

V. CONCLUSION AND FUTURE WORK

In this study, we developed a combination of sketch inversion and colorization DCNN models that could generate photos from hand-drawn sketches with color correction. The robustness of our models could be improved using data augmentation techniques. Thus far, our models work well for dataset under uncontrolled conditions such as variations in scale, position, shape, rotation and noise but we still have many tasks to go in order to generate photos perfectly similar to the actual ones. Ultimately, we would like to use very large Indonesian dataset and more variations in lighting, the use of accessories, and perspective e.g. side face, worm's eye view and bird's eye view. We also foresee the use of adversarial networks for enlarging our dataset and improving our model performance.

REFERENCES

[1] N. Wang, D. Tao, X. Gao, X. Li, & J. Li, "A comprehensive survey to face hallucination," *International Journal of Computer Vision*, vol. 106, pp. 9-30, Aug 2013.

[2] X. Wang and X. Tang, "Face photo-sketch synthesis and recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 31, pp. 1955-1967, Nov 2009.

[3] N. Wang, D. Tao, X. Gao, X. Li, and J. Li, "Transductive face sketch-photo synthesis," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, pp. 1364-1376, Sep 2013.

[4] L. Zhang, L. Lin, X. Wu, S. Ding, and L. Zhang, "End-to-end photo-sketch generation via fully convolutional representation learning," in *International Conference on Multimedia Retrieval*, Association for Computing Machinery (ACM), 2015.

[5] E. Simo-Serra, S. Iizuka, K. Sasaki, and H. Ishikawa, "Learning to simplify: Fully convolutional networks for rough sketch cleanup," *ACM Transactions on Graphics*, vol. 35, no. 4, 2016.

[6] Y. Güçlütürk, U. Güçlü, R. van Lier, and M. A. J. van Gerven, "Convolutional Sketch Inversion," *CoRR*, vol. abs/1606.03073, 2016.

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[8] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Let there be color!: Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification," *ACM Transactions on Graphics*, vol. 35, no. 4, 2016.

[9] B. Xu, "End-to-End Neural Art with Generative Model", 2016. [Online]. Available: http://dmlc.ml/mxnet/2016/06/20/end-to-end-neural-style. [Accessed: 15-July-2016].

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems*, pp. 1097-1105, 2012.

[11] O. M. Parkhi, A. Vedaldi, A. Zisserman, "Deep Face Recognition," in *British Machine Vision Conference*, 2015.

[12] A. Mahendran and A. Vedaldi, "Understanding Deep Image Representations by Inverting Them," in *CVPR*, 2015.

[13] J. B. Martens and L. Meesters, "Image dissimilarity," in *Signal Processing*, vol. 75, pp. 155-176, 1998.

[14] VQEG, "Final Report from the Video Quality Experts Group on the Validation of Objective Models of Video Quality Assesment", 2000. [Online]. Available: http://www.vqeg.org/. [Accessed: 25-Nov-2016].

[15] Z. Wang and A. C. Bovik, "A universal image quality index," *IEEE Transaction on Signal Processing Letters*, vol. 9, pp 81-84, 2002.

[16] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity," *IEEE Transaction on Image Processing*, vol. 13, no. 4, pp 600-612, 2004.