Source Camera Style-Transfer GAN: towards a Better Understanding of Source Camera Identification

Jingtao Li, Yan Xiong, Xing Chen

Abstract

Latent style classification problem such as source camera identification (SCI) is hard even for human. Previous CNN based methods work well but unable provide explanation. In this work, we intend to extract the style information of SCI dataset to interpret the style information. The content is mixed with style in the data which adds difficulty to the extraction. We propose a style-transfer Generative Adversarial Network (GAN) based method to extract the source camera style. And then the style is separated and analyzed. FFT results on the style information shows the frequency feature is a key factor in the SCI task.

1. Introduction

Neural networks have fertilized many real-world applications. Source camera identification (SCI), which is to determine the exact camera model used for capturing an image, has also been tackled well by using convolution neural networks (CNN) [1, 2]. The test accuracy using neural networks beats several conventional feature extraction methods such as SVM. However, the style information in the trained CNN, unlike content information, is particularly difficult to perceive even for human. To understand why CNN can perform well in such a style-classification problem, is important to SCI study and explainable AI.

The reverse process of CNN, Generative Adversarial Network (GAN) [3] shows great potential to preserve style information. Recently, Style-Transfer GAN [4] was proposed to forge style-transfer of any given image to a style class. A robust style-transfer GAN is meant to be content-free, thus provide us a way to separate the latent style to a parametric model to investigate.

We propose cameraGAN to preserve the style of a target camera model by using GAN. We adopt style-transfer GAN [5] as our algorithm base. The workflow is as following: first, we train a CNN on the SCI dataset and uses it as the auxiliary classifier in training. Second, the generator model is trained using CIFAR-10 image as no-style image such that the generated fake style imgae can fool the auxiliary classifier consistently. Also, we uses another classifier to testify the performance. The fake style images are studied towards a better understanding. Our main contribution can be summarized as:

- We present a method to provide explanation for CNN on style classification task. While three state-of-art CNN visualization methods all fail, this method achieves a reasonable explanation.
- We found the SCI CNN model depends on the frequency feature particularly, which is in consistency with the established domain knowledge. We strongly believe this method could possibly be used to extract domain knowledge from big data and generalized to other tasks.

The rest of the paper is organized as follows: In section II, we provide the background on SCI and recent works towards explainable neural networks. In section III, we present the style-transfer GAN method. In section IV, we present experimental details. In section V, we conclude our findings. We include multiple figures in appendix.

2. Background

2.1. Previous work on Source Camera Identification

The physics aspects of the camera are firstly investigated. Choi et al. [6] found the inherent radial distortions serve as unique fingerprints in the images. Chennamma and Rangarajan [7] investigate the readout noise of the image sensor.

Feature vector extraction technique are also widely used. Filler et al. [8] defined the sensor photo-response non-uniformity (PRNU) as the main component of a camera fingerprint and their classification process achieved an error rate of 11.2%. Kharrazi et al. [9] developed a supervised learning approach based on features extracted from both the spatial domain and the wavelet domain, obtaining error rates between 5% and 22%. Very recently, using deep neural network shows better performance on accuracy. Tuama et al. [1] uses modified GoogleNet and achieves an accuracy of 98.99% on a 33 category dataset.

Recently, domain knowledge is combined with deep learning methods. Among the recent papers, [10, 11] combines the previously established noise pattern with the neural networks. In [10], a high-pass filter (HPF) and fast fourier transformation (FFT) is used in preprocessing. The frequency pattern of noise pattern of images captured by digital camera was shown to be consistent for a group of images. They reported the best ever performance (nearly 100% accuracy).

2.2. Previous work towards neural network understanding

While machine learning models and deep neural networks (DNN) are widely used in image related areas and achieve state-of-the-art performance in various tasks, it is also important to understand why these models have such capability.

Visualization of DNN is another common topic in DNN understanding. [12] considered two visualization techniques based on the gradient computation of class score and input images. The first one generated the image that has the maximum class score and the other computes the class saliency map that is specific to a given image and class. These maps are shown to contain critical information in classification DNNs and are useful in weakly supervised object segmentation. Besides, the visualization of activation values produced by intermediate layers is also found interesting in [13]. This method provides recognizable images that unveil the feature-level patterns which help DNN make the classification.

Recent works in DNN understanding keep exploring from various aspects. [14] design a comprehensive and interactive system that visualizes the output of recurrent neural networks (RNN) and explains the information behind the time-sequential weather forecasting analysis. [15] and [16] provide further developed techniques to understanding convolutional neural network (CNN). The prediction difference analysis, proposed in [15], visualizes the response of a deep neural network to a specific input and provides insight into the decision-making process of classifiers. [16] explains the CNN elements in functionality and efficiency and provides a deep understanding of how CNN achieves superior performance in the image processing area. In [17], Lime, a novel explanation technique, is proposed to explain the prediction of the classifier by learning an interpretable model. Based on the model, it provides valid evidence when deciding if a classification can be trusted.

3. Experimental Settings

3.1. Platform

All experiments are conducted on a machine equipped with an NVIDIA RTX 2080ti GPU. The training and inference for the CNN and GAN models are implemented using Pytorch.

3.2. Datasets

For the dataset, we started with a four-class toy dataset provided by a kaggle competition come from four camera models: HTC-1-M7, iPhone-6, LG-Nexus-5x and Samsung Galaxy Note3. Each category has 275 different images, a randomly picked image sample is shown in Figure 1. The content of the image is not controlled which means 275 pictures are taken randomly by each mobile phone camera). For the ease of fast training for this project, we partition the 275 images into 264 training and 11 testing images per class. Then, we apply a random crop of 30 sub-images of size 32x32 on each of the original partitions to expand the dataset size by 30 times. In the future work, we are to apply the same method on Dresden database [18], which is widely used as benchmark in recent works [10, 11].

4. Proposed Method

GAN is proposed as a min-max optimization problem in [3]. It consists of a generator model and a discriminator model. The training of GAN can beseen as two interleaved steps corresponding to the min-max optimization of each model. The generator can learn the distribution that hidden in large amount of data and forges realistic images. GAN serves as the essential component of this work.

4.1. CNN classifier

A CNN auxiliary classifier to perform the classification task is described in this section, which will contribute to loss term in the GAN training. We selected Resnet-20 [19] for its outstanding performance in image classfication tasks. For the dataset, we uses the cropped version of the the four-class toy dataset. The cropped dataset consists of 31,680 training images and 1320 testing images of 4 categories. Additionally, we apply normalization and random horizontal flipping which are standard augmentation techniques.

For the training settings, we use cross-entropy loss and Adam optimizer with a 0.001 learning rate with decay. The batch size is 64 and we train for a total of 50 epochs.

A test CNN classifier which takes a quantized version of Resnet-20 with 8-bit weights. We keep the same training data and settings as auxiliary classifier. The intention to have a test CNN model is to verify the consistence of the generator at the end of GAN training.

4.2. Style-transfer GAN

For the GAN training architecture, we adopt the same architecture as stated in gated-GAN [5].

The encoder, gated-transformer and decoder together makes the generator: The encoder which is made of 3 convolutional layers, acts as a basic feature extractor. The

(a) HTC M7 (b) iPhone 6 (c) LG Nexus 5x (d) Samsung Galaxy Note 3

Figure 1: A visualization of selected toy dataset.

transformer consist of a residual block and a control unit to select the style. In the original gated-GAN, the control unit is used to separate different style transformer and different style transfer tasks share the same encoder and decoder. While in our work, the 4 different style transfer tasks have totally different generator, that is, every style transfer task has its own GAN. The decoder consists of 4 residual blocks, 2 transpose convolutional layers, and 1 convolutional layer. Here, the output of decoder is regularized by a loss term which calculated the L1 distance from the original input to keep the content. Additionally, a TV loss is used to improve the smooth and visualization of the generated images.

As mentioned, in a typical GAN training, the discriminator must be trained from scratch with the generator to stabilize the training. Discriminator has a symmetrical structure with generator which has 4 convolutional layers and a fully connected layer for classification. The job of discriminator is to distinguish real style image (directly drawn from dataset) and fake images (generated by the generator) are taken as input and contribute as the adversarial loss. Additionally, the fake images are also fed into the auxiliary classifier to tests its real performance on a well-trained CNN, which contributes to the cross-entropy loss.

5. Experimental results

5.1. Auxiliary & Test classifier

For the auxiliary classifier and the test classifier, after 50 epochs of training, we achieve 92.6% and 93.18% accuracy on the test data, respectively.

5.2. The difficulty of understanding latent style

To address the difficulty of understanding the neural network's behavior of camera model classification, we first try several state-of-art model visualization technique on our resnet-20 model, which are saliency map [12], filter explanation [13] and LIME [17].

The saliency map is based on calculating the gradients for a set of inputs which are randomly sampled. The backward gradient calculation is done till the input layer to show the pixel-wise importance. We sample 16 images from four different classes, the saliency maps are shown in Figure 3 (see Appendix). We observe a major difference from a common content-based classification task. Our saliency maps indicate the model tries to avoid the edge and is more interested in the smooth area.

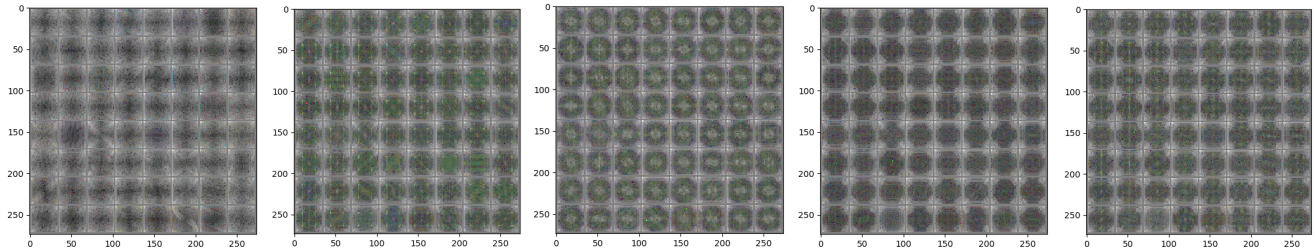
Next, we apply the method in [13], which the filter relative importance is extracted as shown in Figure 4 (see Appendix). Similar to the content-based classifier this time, We can see some of the filter is trying to recognize the edge, which is more content-related. Combine both two observation so far, the mixture of information makes it difficult to decide the true behavior of the CNN model.

Additionally, we use the state-of-the-art NN explanation tool, LIME. It uses a linear model to approximate locally on some given inputs to figure out which segments in these inputs are going to influence classification the most. Based on this, the LIME tool would mark the region as green or red to indicate the agree/disagree to the target class. We apply LIME to our model and choose the class to be "Samsung Galaxy Note 3". The LIME region sensitivity is shown in Figure 5 (see Appendix). The LIME visualization effect is even more disappointing, shows no sign of what it is doing.

5.3. Style-transfer GAN

For the training of style-transfer GAN, CIFAR-10 training set is used as no-style inputs of the generator and the SCI dataset we used in the CNN is used as real style image feeding to the discriminator. For different camera classes, we reinitialize the generator/discriminator part every time training a new generator of a camera style. All four classes converges successfully and the training accuracy all reaches 100%.

After training, we perform the style transfer using each



(a) no-style input (FFT) (b) generated class 1 (FFT) (c) generated class 2 (FFT) (d) generated class 3 (FFT) (e) generated class 4 (FFT)

Figure 2: Visualization of FFT on generated images. We observe consistent frequency patterns are observed across different content of a given class, which indicates the successful extraction of content-free information.

generator on the CIFAR-10 test data to verify its consistence as validation. Results are shown in table I. We also use the test classfier to testify the generator's performance and also demonstrate its the transfer ability.

Table 1: Test accuracy of pre-trained classifier on images generated by four classes' generator using CIFAR-10 test data, the label for generated image is fixed to the target class.

Class	Test Accuracy	Transfer Accuracy
1	97.87	65.48
2	99.48	97.85
3	99.65	99.75
4	62.56	54.85

It is shown the generator of class 2 and 3 can consistently fool the auxiliary model with high test accuracy and transfer well to the test classifier. For class 1, the generator can fool auxiliary model but accuracy drops on the test classifier. For class 4, the generator performs not so good with 62.56% accuracy of its generated image being considered as class 4 by the classifier. Despite the reason for some of the imperfections, all four generator achieve much higher accuracy than a random guess of 25%.

5.4. Deep understanding of the SCI problem using style-transfer GAN

We separate the generator for investigation in this subsection. First, we feed a random 64 samples of CIFAR-10 test image as no-style input images to the four generator and have a direct visualization on it. As shown in Figure 6 (see Appendix), the content of generated images can still be easily recognized. We observed two noticeable difference. The first is comparing to the original image, generated image has a different color and varies from different classes. The second is we notice some artifacts that also vary from class to class.

To investigate the color difference, we construct some pure color inputs and feed them to different generators. Surprisingly, as shown in Figure 7 (see Appendix), different generator has almost zero response to a pure colored image. This observation indicates the color difference is not the correct explanation of the CNN model.

The noise pattern in the generated images drive us to investigate in the frequency domain. We do Fast Fourier Transformation (FFT) on each channel of the 64 samples of CIFAR-10 images generated by each generator. The FFT of each channel (RGB image) is computed and concatenate back have the same channel size as original image. The value is magnified by taking its dB form for better visualization. The results are shown in Figure 2, we observe a consistent pattern across different contents of a given class, which indicates the pattern is content-free. In other words, a certain frequency pattern is added by the generator to the no-style inputs to successfully forge the target style. It clearly indicates the CNN classifier is based on frequency feature to make prediction so that a GAN that forges such features can consistently fool it. The finding is consistent with prior knowledge in the SCI task [10, 11], which also suggests the frequency should be addressed to train a good CNN classifier on SCI tasks.

Inspired by [10], We further perform HPF and FFT directly on original SCI training images in Figure 8 (see appendix), which does not clearly show the consistent pattern as we observe in Figure 2.

6. Future works

We will perform what we did on a toy dataset on Dresden dataset. And We will revisit the work as a domain knowledge extraction tool, to see whether this is applicable for other tasks.

References

[1] A. Tuama, F. Comby, and M. Chaumont, "Camera model identification with the use of deep convolutional neural networks," in *2016 IEEE International workshop on information forensics and security (WIFS)*. IEEE, 2016, pp. 1–6. 1

[2] D. Freire-Obregón, F. Narducci, S. Barra, and M. Castrillón-Santana, "Deep learning for source camera identification on mobile devices," *Pattern Recognition Letters*, vol. 126, pp. 86–91, 2019. 1

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680. 1, 2

[4] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song, "Neural style transfer: A review," *IEEE transactions on visualization and computer graphics*, 2019. 1

[5] X. Chen, C. Xu, X. Yang, L. Song, and D. Tao, "Gated-gan: Adversarial gated networks for multi-collection style transfer," *IEEE Transactions on Image Processing*, vol. 28, no. 2, pp. 546–560, 2018. 1, 2

[6] K. San Choi, E. Y. Lam, and K. K. Wong, "Automatic source camera identification using the intrinsic lens radial distortion," *Optics express*, vol. 14, no. 24, pp. 11 551–11 565, 2006. 1

[7] H. Chennamma and L. Rangarajan, "Source camera identification based on sensor readout noise," *International Journal of Digital Crime and Forensics (IJDCF)*, vol. 2, no. 3, pp. 28–42, 2010. 1

[8] T. Filler, J. Fridrich, and M. Goljan, "Using sensor pattern noise for camera model identification," in *2008 15th IEEE International Conference on Image Processing*. IEEE, 2008, pp. 1296–1299. 1

[9] M. Kharrazi, H. T. Sencar, and N. Memon, "Blind source camera identification," in *2004 International Conference on Image Processing, 2004. ICIP'04.*, vol. 1. IEEE, 2004, pp. 709–712. 1

[10] T. Cai, Z. Shao, Y. Tomioka, Y. Liu, and Z. Li, "Cnn-based camera model identification using image noise in frequency domain," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*. IEEE, 2019, pp. 3518–3524. 2, 4

[11] X. Ding, Y. Chen, Z. Tang, and Y. Huang, "Camera identification based on domain knowledge-driven deep multi-task learning," *IEEE Access*, vol. 7, pp. 25 878–25 890, 2019. 2, 4

[12] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013. 2, 3

[13] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," *arXiv preprint arXiv:1506.06579*, 2015. 2, 3

[14] I. Roesch and T. Günther, "Visualization of neural network predictions for weather forecasting," in *Computer Graphics Forum*, vol. 38, no. 1. Wiley Online Library, 2019, pp. 209–220. 2

[15] L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling, "Visualizing deep neural network decisions: Prediction difference analysis," *arXiv preprint arXiv:1702.04595*, 2017. 2

[16] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 International Conference on Engineering and Technology (ICET)*. IEEE, 2017, pp. 1–6. 2

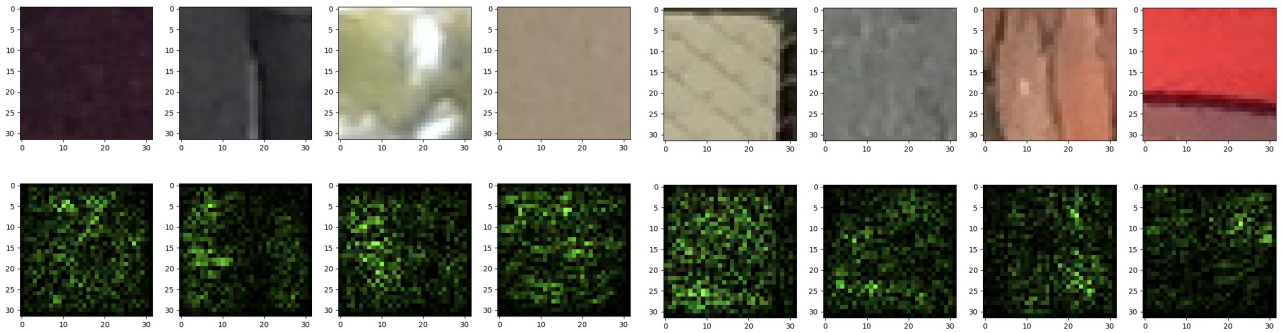
[17] M. T. Ribeiro, S. Singh, and C. Guestrin, "" why should i trust you?" explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144. 2, 3

[18] T. Gloe and R. Böhme, "The dresden image database for benchmarking digital image forensics," *Journal of Digital Forensic Practice*, vol. 3, no. 2-4, pp. 150–159, 2010. 2

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778. 2

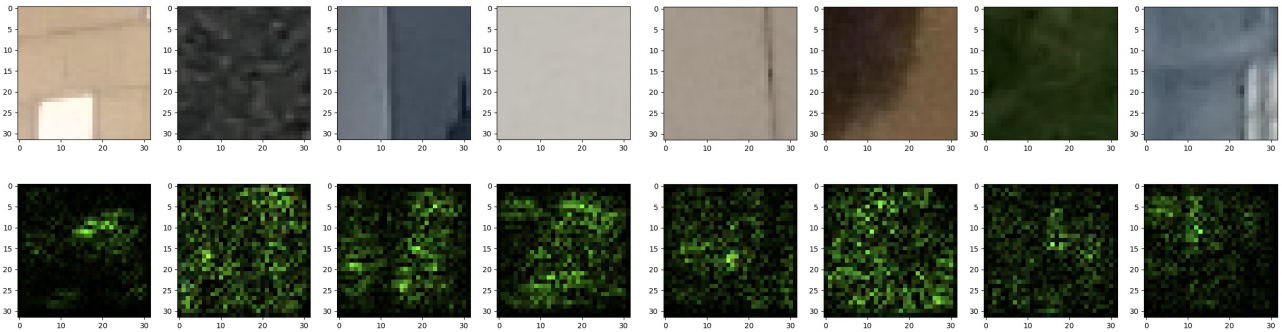
Appendices

A. Appended Graphs



(a) HTC M7

(b) iPhone 6



(c) LG Nexus 5x

(d) Samsung Galaxy Note 3

Figure 3: A visualization of saliency maps of different category. The saliency maps indicate the model tries to avoid the edge and is more interested in the smooth area.

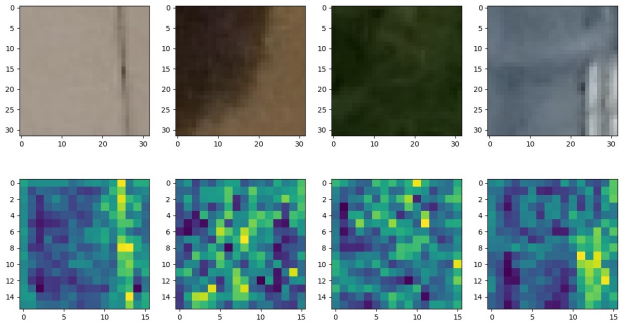


Figure 4: Filter explanation. Some of the filter is trying to recognize the edge, which could focus on the content.

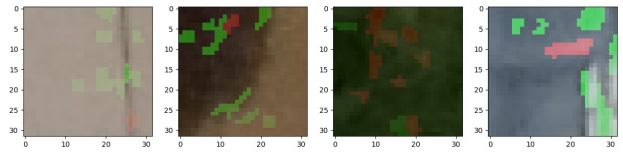
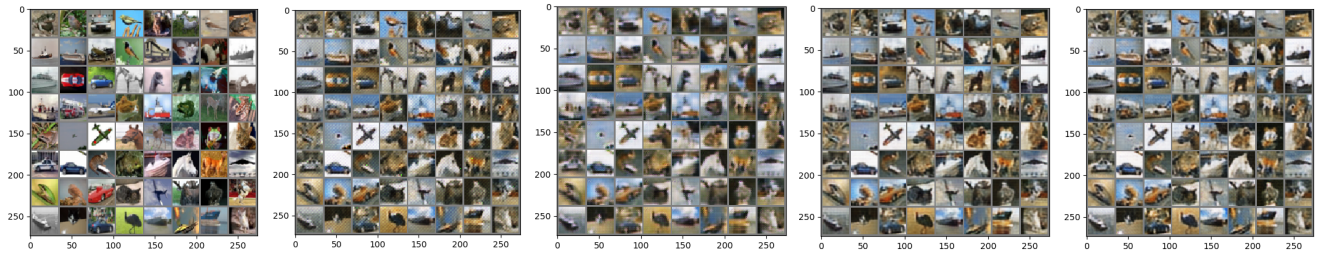


Figure 5: LIME explanation. It shows random behavior.

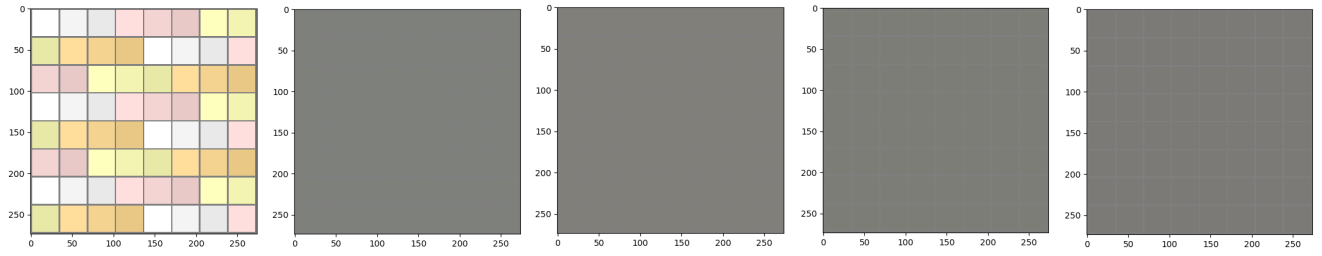


(a) no-style input (8x8) (b) generated class 1 (8x8) (c) generated class 2 (8x8) (d) generated class 3 (8x8) (e) generated class 4 (8x8)

(f) no-style input image (plane) (g) generated class 1 image (plane) (h) generated class 2 image (plane) (i) generated class 3 image (plane) (j) generated class 4 image (plane)

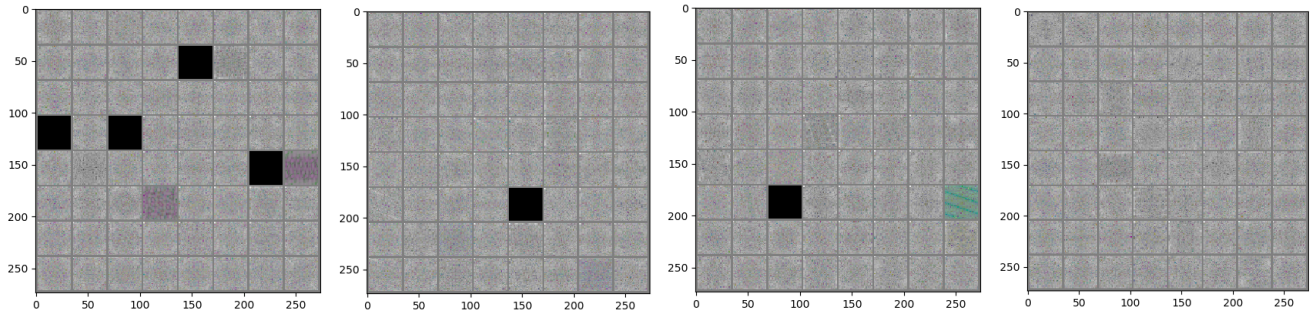
(k) no-style input image (frog) (l) generated class 1 image (frog) (m) generated class 2 image (frog) (n) generated class 3 image (frog) (o) generated class 4 image (frog)

Figure 6: A direct visualization of generated style versus original no-style input. We can observe not only the color got changed, the artifacts of images generated by different generator seems to have an unique pattern.



(a) pure color input (b) generated class 1 (c) generated class 2 (d) generated class 3 (e) generated class 4

Figure 7: Pure color image style-transfer experiments. They are all transformed to a gray image with almost 0 pixel value, indicating the color difference is neglected by the generator.



(a) original class 1 (HPF+FFT) (b) original class 2 (HPF+FFT) (c) original class 3 (HPF+FFT) (d) original class 4 (HPF+FFT)

Figure 8: Visualization of HPF + FFT on original training images. We cannot observe clear frequency patterns across different class.