Use GAN to Draw Cubism and Impressionism Paintings

Chengyi Zhang
Peter Xie

Background

“Computers are useless.They can only give you answers” -- Picasso

Paintings are a huge part of our life. (decoration, presents, online profile pictures/backgrounds, etc)

Copyright issues, can’t find the paintings you like, want to create your own paintings but don’t have enough skills …

Solution: Generative adversarial networks

Data processing

Source: Wikiart

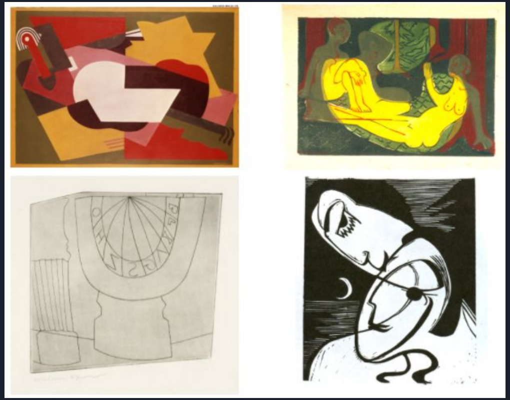
Cubism- 2,235 paintings in total

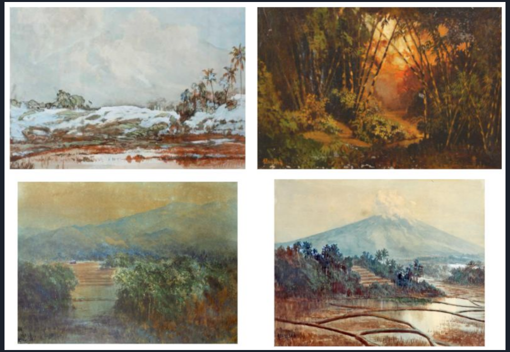
Impressionism-13,060 paintings in total

Resize into 128*128

Save as numpy arrays.

Normalization.





GAN

Generator and discriminator

Generator takes in a seed (a vector) and outputs an image. Discriminator takes in images and determine whether it’s fake or real.

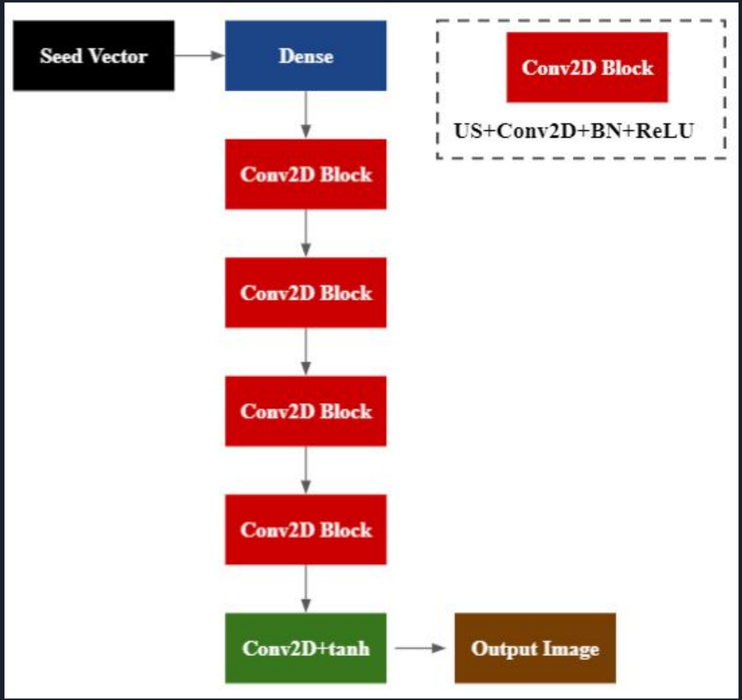
Discriminator trying to be better at classifying, while generator trying to fool the discriminator.

Loss function: Cross entropy

We used Google Colab to train the neural network.

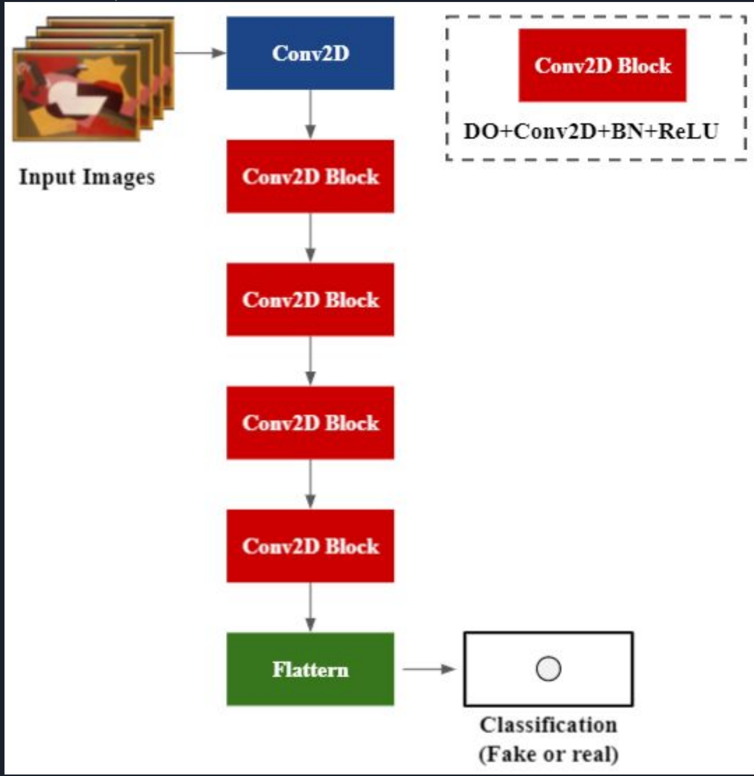
$$H(p,q) = -\sum_i p_i \log(q_i)$$

Generator



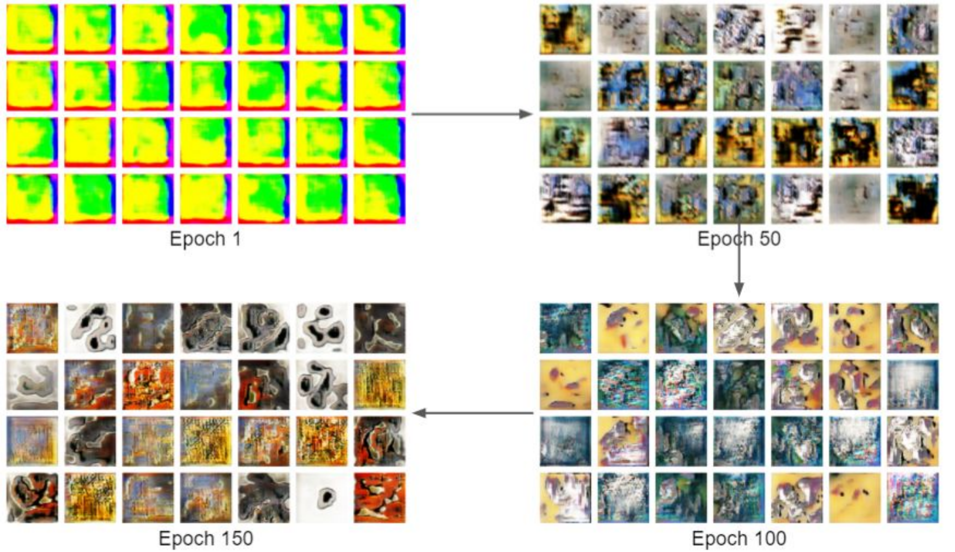
Layer (type)	Output Shape	Param #
dense (Dense)	(None, 4096)	413696
reshape (Reshape)	(None, 4, 4, 256)	0
up_sampling2d (UpSampling2D)	(None, 8, 8, 256)	0
conv2d (Conv2D)	(None, 8, 8, 256)	590080
batch_normalization (BatchNo	(None, 8, 8, 256)	1024
activation (Activation)	(None, 8, 8, 256)	0
up_sampling2d_1 (UpSampling2	(None, 16, 16, 256)	0
conv2d_1 (Conv2D)	(None, 16, 16, 256)	590080
batch_normalization_1 (Batch	(None, 16, 16, 256)	1024
activation_1 (Activation)	(None, 16, 16, 256)	0
up_sampling2d_2 (UpSampling2	(None, 32, 32, 256)	0
conv2d_2 (Conv2D)	(None, 32, 32, 128)	295040
batch_normalization_2 (Batch	(None, 32, 32, 128)	512
activation_2 (Activation)	(None, 32, 32, 128)	0
up_sampling2d_3 (UpSampling2	(None, 128, 128, 128)	0
conv2d_3 (Conv2D)	(None, 128, 128, 128)	147584
batch_normalization_3 (Batch	(None, 128, 128, 128)	512
activation_3 (Activation)	(None, 128, 128, 128)	0
conv2d_4 (Conv2D)	(None, 128, 128, 3)	3459
activation_4 (Activation)	(None, 128, 128, 3)	0

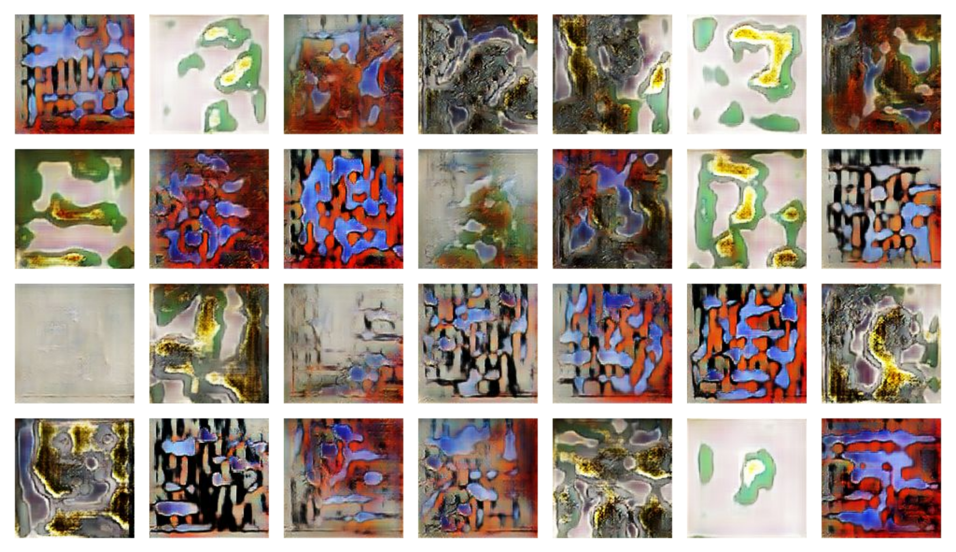
Discriminator



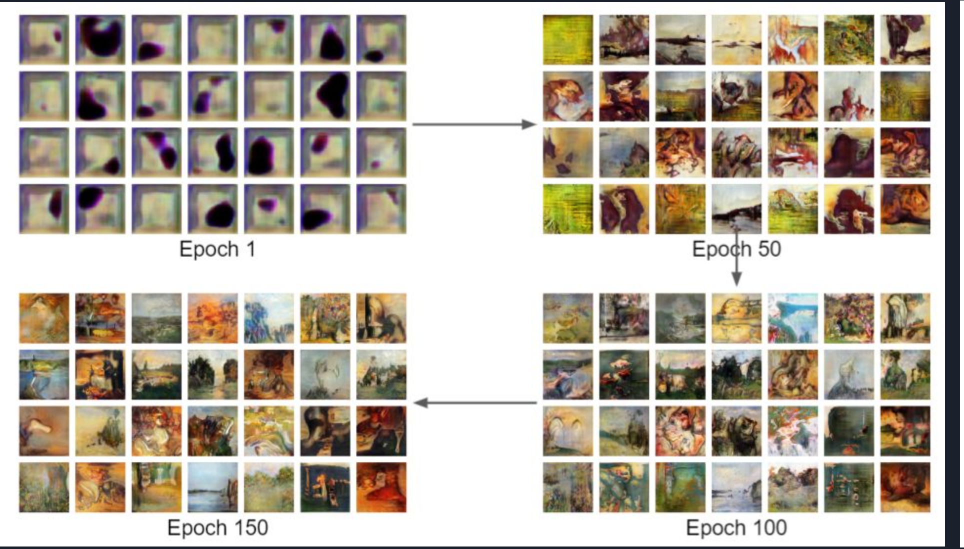
Layer (type)	Output Shape	Param #
conv2d_5 (Conv2D)	(None, 64, 64, 32)	896
leaky_re_lu (LeakyReLU)	(None, 64, 64, 32)	0
dropout (Dropout)	(None, 64, 64, 32)	0
conv2d_6 (Conv2D)	(None, 32, 32, 64)	18496
zero_padding2d (ZeroPadding2	(None, 33, 33, 64)	0
batch_normalization_4 (Batch	(None, 33, 33, 64)	256
leaky_re_lu_1 (LeakyReLU)	(None, 33, 33, 64)	0
dropout_1 (Dropout)	(None, 33, 33, 64)	0
conv2d_7 (Conv2D)	(None, 17, 17, 128)	73856
batch_normalization_5 (Batch	(None, 17, 17, 128)	512
leaky_re_lu_2 (LeakyReLU)	(None, 17, 17, 128)	0
dropout_2 (Dropout)	(None, 17, 17, 128)	0
conv2d_8 (Conv2D)	(None, 17, 17, 256)	295168
batch_normalization_6 (Batch	(None, 17, 17, 256)	1024
leaky_re_lu_3 (LeakyReLU)	(None, 17, 17, 256)	0
dropout_3 (Dropout)	(None, 17, 17, 256)	0
conv2d_9 (Conv2D)	(None, 17, 17, 512)	1180160
batch_normalization_7 (Batch	(None, 17, 17, 512)	2048
leaky_re_lu_4 (LeakyReLU)	(None, 17, 17, 512)	0
dropout_4 (Dropout)	(None, 17, 17, 512)	0
flatten (Flatten)	(None, 147968)	0
dense_1 (Dense)	(None, 1)	147969

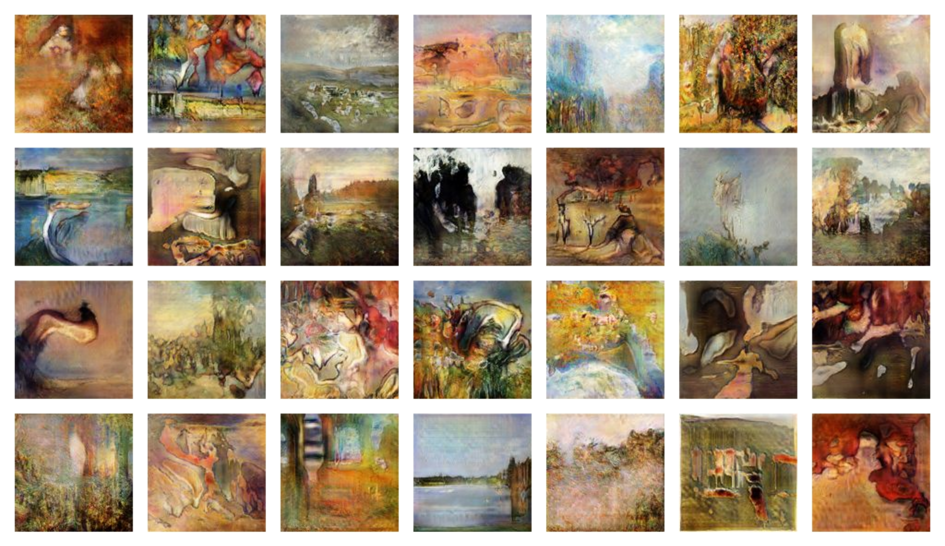
Result for Cubism



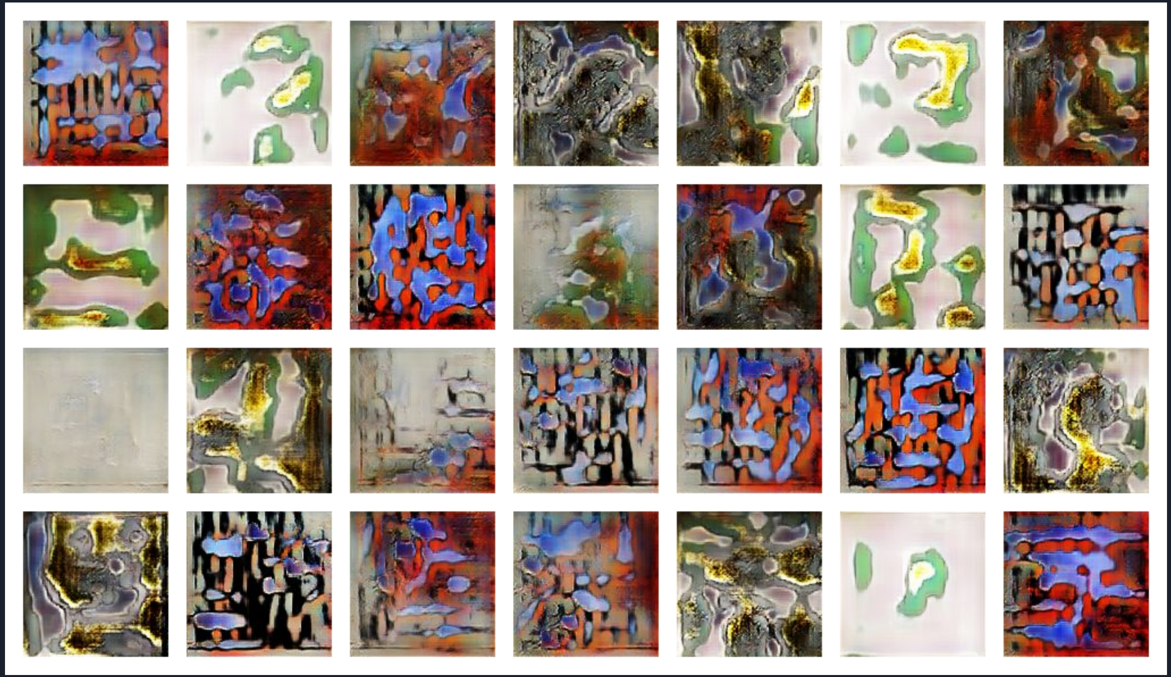


Result for Impressionism

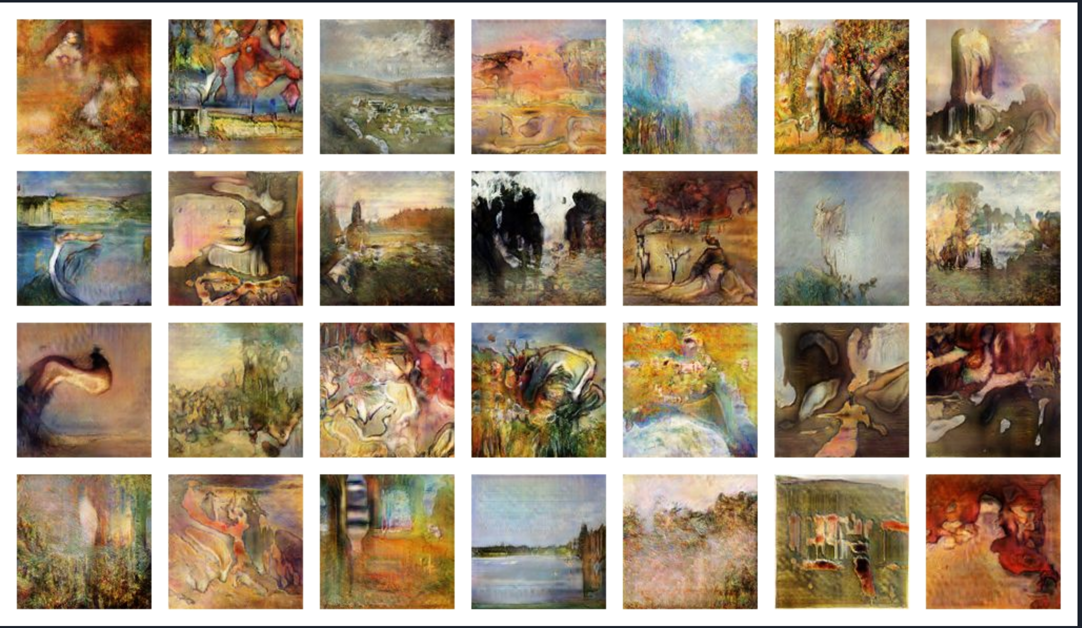




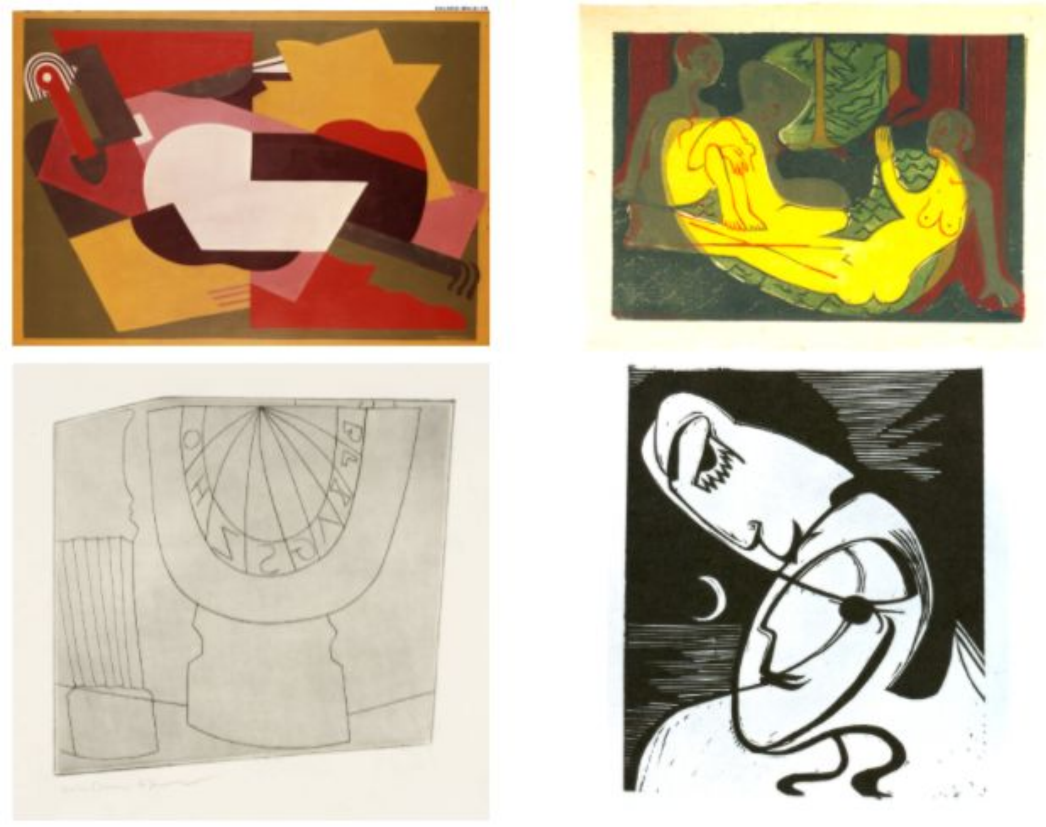
Cubism Comparison



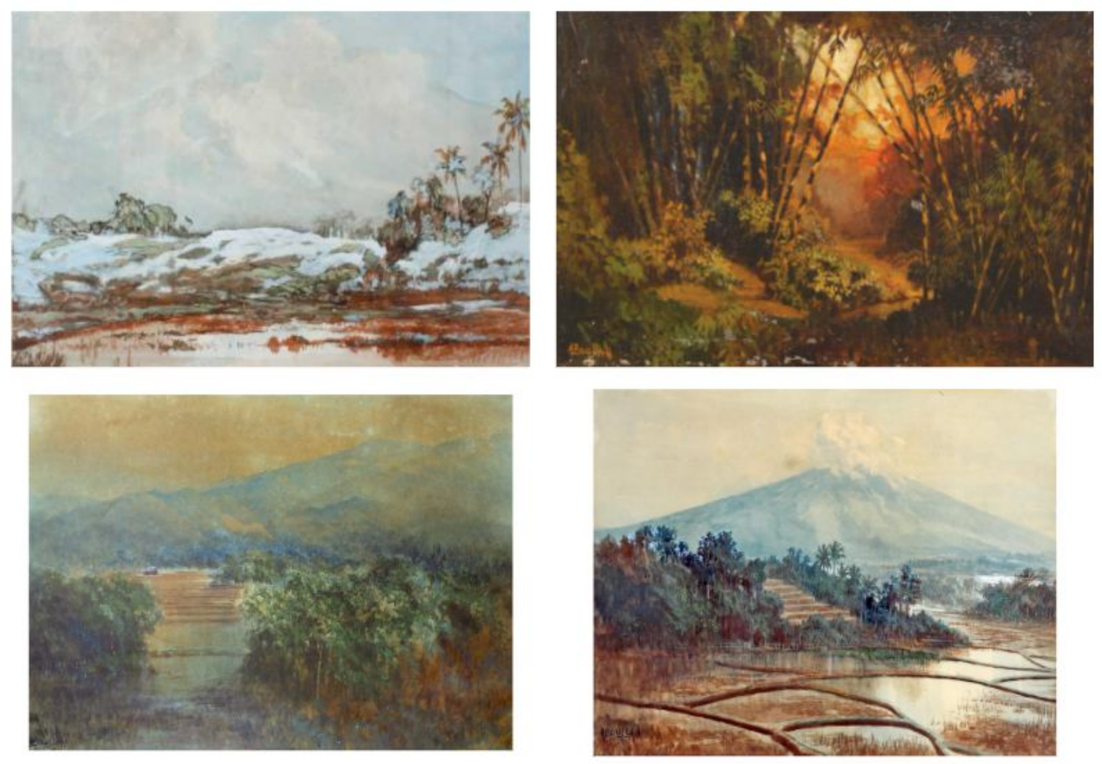
Impressionism comparison

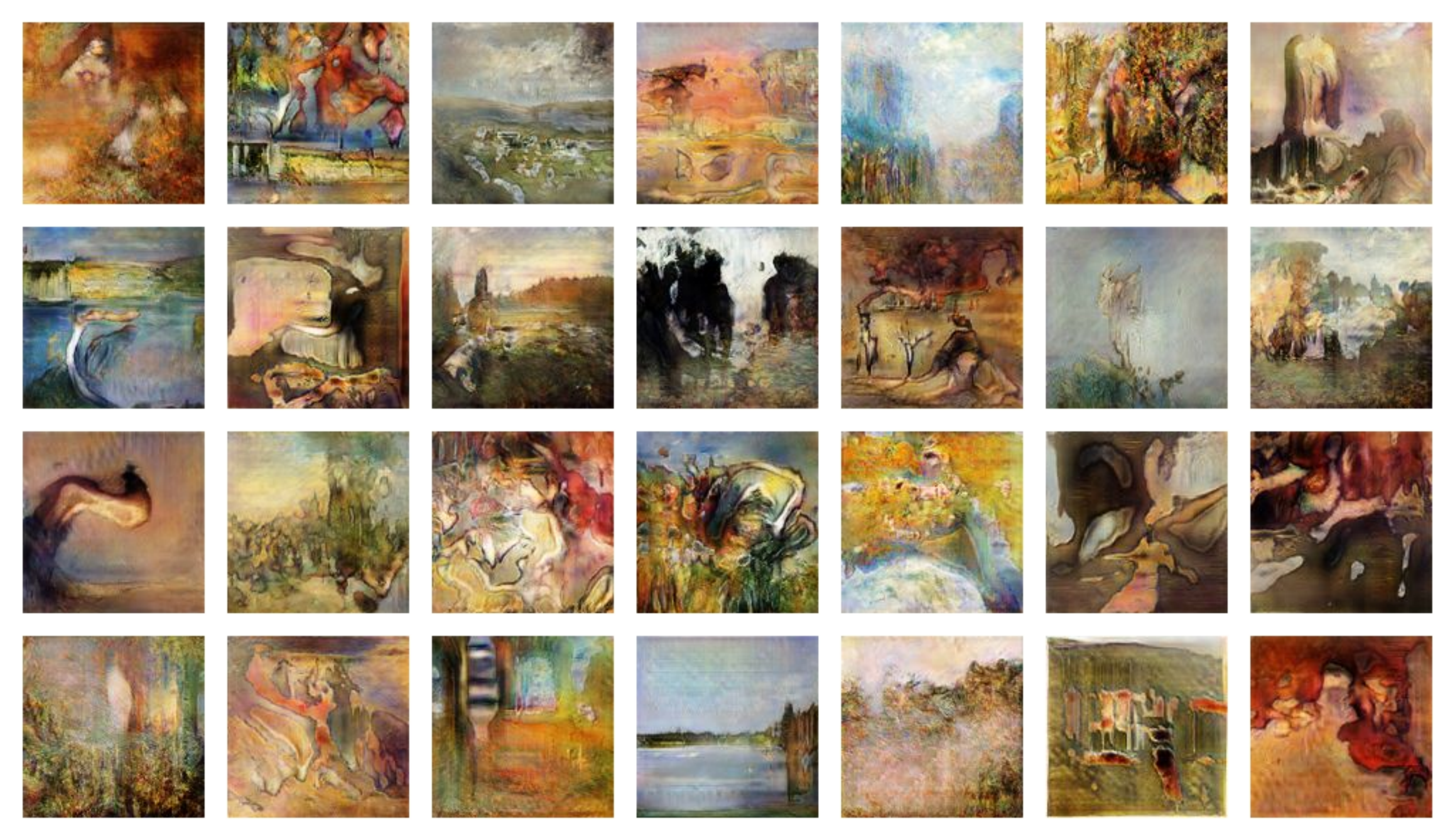


Problems about Cubism inputs



Advantages about Impressionism inputs





Thank you!

Chengyi Zhang

Peter Xie

University of Rochester

CSC 249 Machine Vision

Code: https://github.com/czhang64/CSC249_GAN

Works Cited:

A. Radford, L. Metz, and S. Chintala, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks,” arXiv:1511.06434v2. 2016.

Heaton, Jeff. “Creating and Training a Generative Adversarial Networks (GAN) in Keras (7.2).” *YouTube*, uploaded by Jeff Heaton, 2 July 2019, https://youtu.be/T-MCludVNn4.

K. Jones, “GANGogh: Creating Art with GANs,” Towards Data Science. 2017

T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive Growing of GANs for Improved Quality, Stability, and Variation,” arXiv:1710.10196v3. 2018.