Image Style Transfer using Convolutional Neural Network

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Runyu Gao, Chi Zhang, Kaixuan Zhang

Problem Definition

- We want to let our model recognize the content and style of images.
- And then combine the content of one picture with another picture that has particular style and features, we hope to combine them to produce a new picture that it content is generally unchanged but have new style from style image.
- We build the convolutional neural network for content and style features selection and combination. We must pre-process the data (images) and fit them into Convolutional neural network (CNN), especially VGG19, a pre-trained CNN model.

Why is this an interesting problem? - We wanna be artists!

- **Real-world implications:**

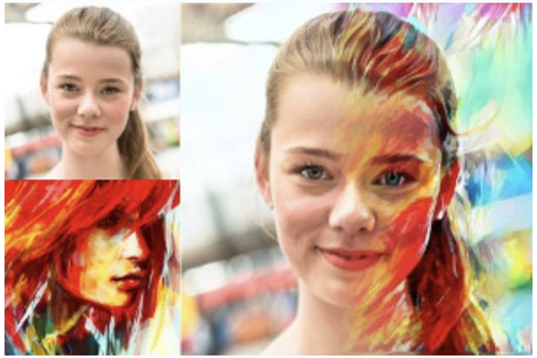
  Some mobile applications with similar function

- **This problem is similar to we've seen in lecture:**

  In deep learning lecture, there has an example of image generation that take an image and a painting, render the image in the style of the painting.

- **Ethical Issues:**

  The pictures being choose should not infringe on other people's portrait rights, resolutely do not use unauthorized and non-public pictures. Also, the generated images should not violate the rights of intellectual property of artists.



Data: Image Content image and style image

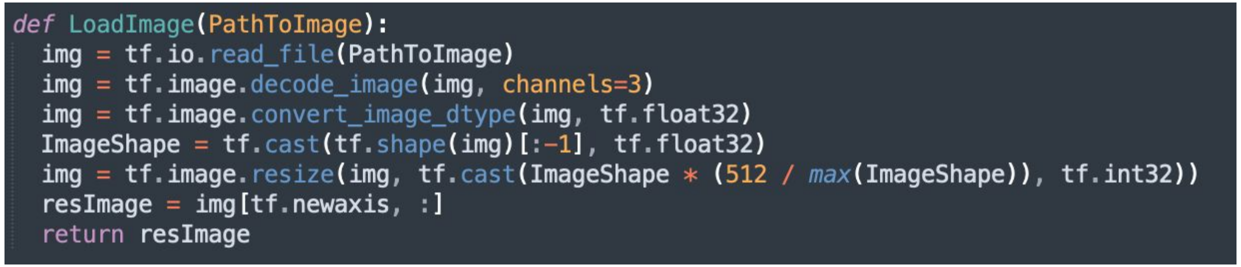
Examples:

Content image + Style image

Content image + Style image

Data: Image Content image and style image

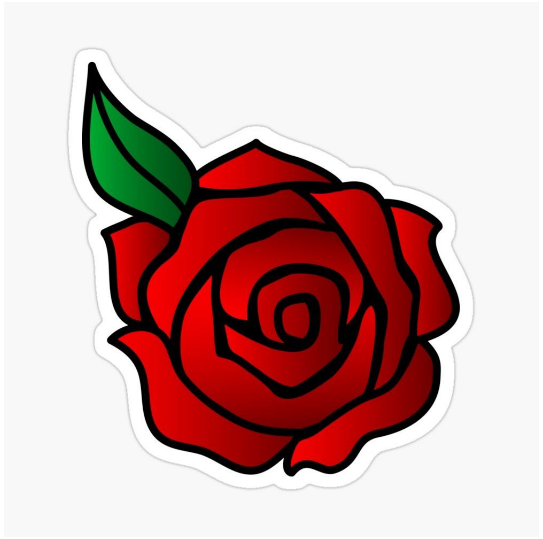
- Because our model is a generative model, we do not need a very large data set. Actually, at least we only need 2 examples: one content image and one style image.
- Pre-processing: For following use and more efficient computation, we have to resize the images to 512 pixels at most, then transfer the images into tensors. Below is our function code to preprocess the image data.



```python
def LoadImage(PathToImage):
  img = tf.io.read_file(PathToImage)
  img = tf.image.decode_image(img, channels=3)
  img = tf.image.convert_image_dtype(img, tf.float32)
  ImageShape = tf.cast(tf.shape(img)[:-1], tf.float32)
  img = tf.image.resize(img, tf.cast(ImageShape * (512 / max(ImageShape)), tf.int32))
  resImage = img[tf.newaxis, :]
  return resImage
```

Baseline:

- At least we should Combine a black and white photo is with a colored cartoon picture to produce a new picture in which the black and white photo is given the color of the cartoon. Because black-white images are the simplest content images with little noises of style. Also, colored cartoon are also simple in style.



Method

- Convolutional neural network (CNN), VGG19, a pre-trained CNN model. For VGG19 could use layers representation from original images in a unified framework for replication and optimization. Also, there are more layers in VGG19 than other pre-trained CNN models, thus our generated image would be more accurate and clear.
- The first few layers represent low-level features such as edges and textures. As the layers deepen, the last layers represent higher-level features: parts of entities, such as wheels or eyes.
- The style of an image can be described by the mean value and correlation on different feature maps. By computing the outer product of the feature vectors at each position and averaging this outer product over all positions, a Gramian Matrix containing this information can be computed. The Gramian Matrix for a particular layer is calculated as follows:
- We could use the Gramian Matrix to represent the style and mathematically.

$$G^l_{cd} = \frac{\sum_{ij} F^l_{ijc}(x) F^l_{ijd}(x)}{IJ}$$

Architecture of VGG19

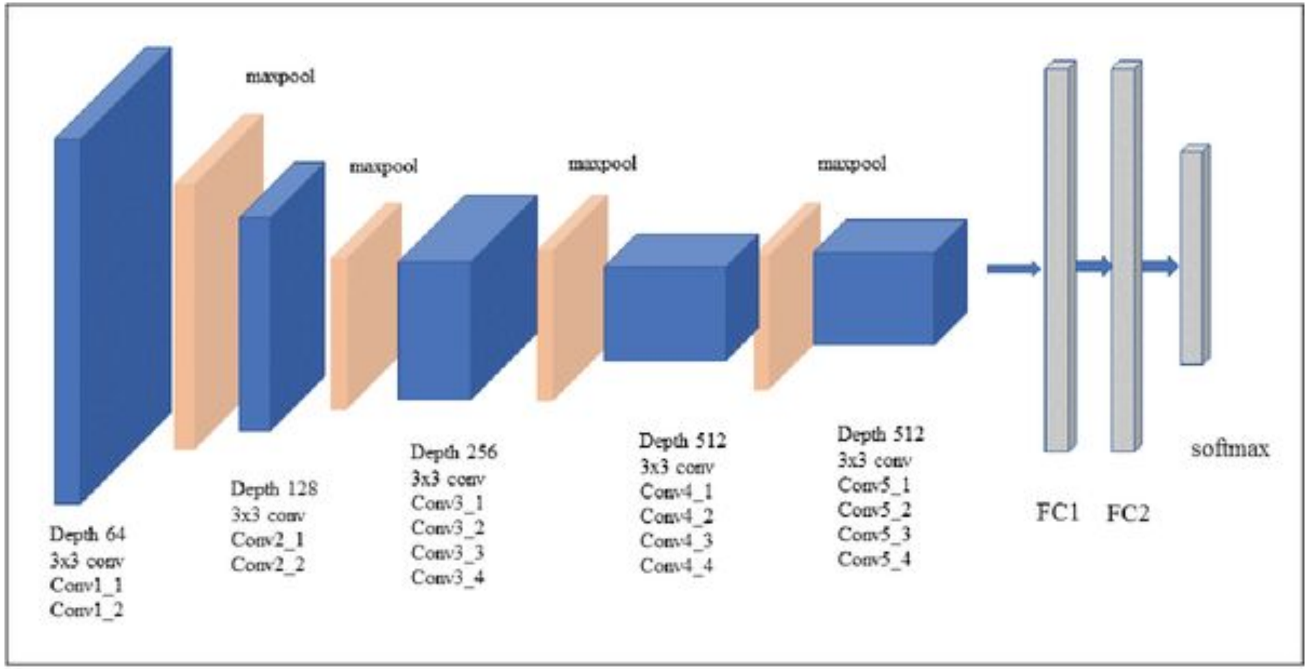
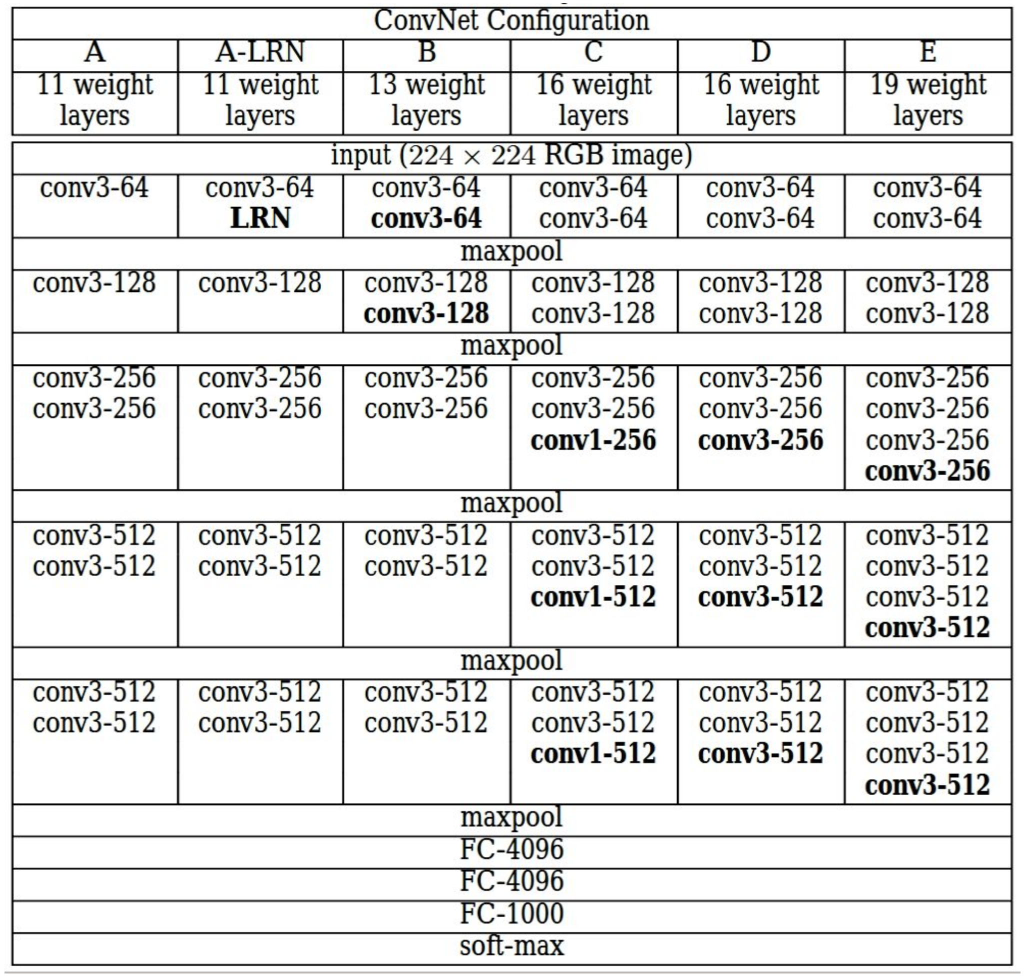


Fig. 3. VGG-19 network architecture

Architecture

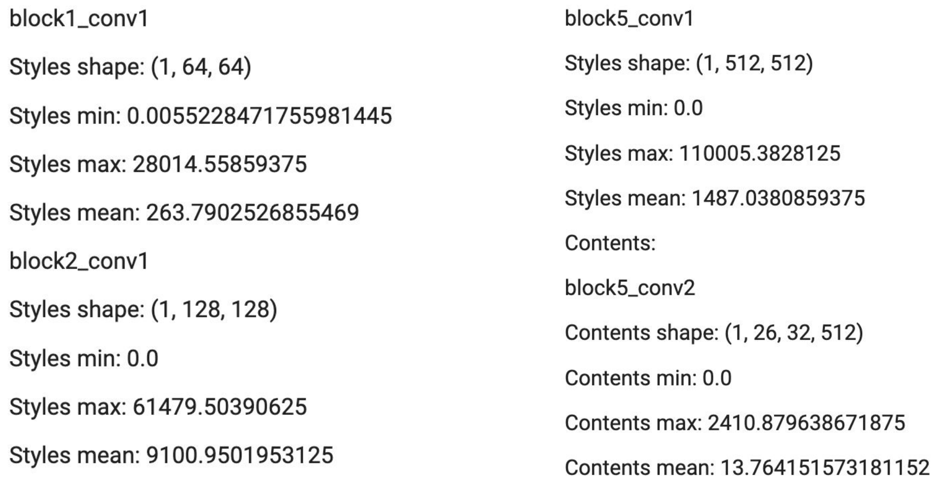
# of VGG19



ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Training Process:

- Extracts the features of the input image through **learning filters**, so as to obtain different feature maps at each layer. If we adopt the trained convolutional neural network to identify the objects in the image (such as style and color), then the network will build, output, and containing the similar features as the given image content and style. We could evaluate it from the loss function.
- Part of the gramian matrix of style_layers and content layers:

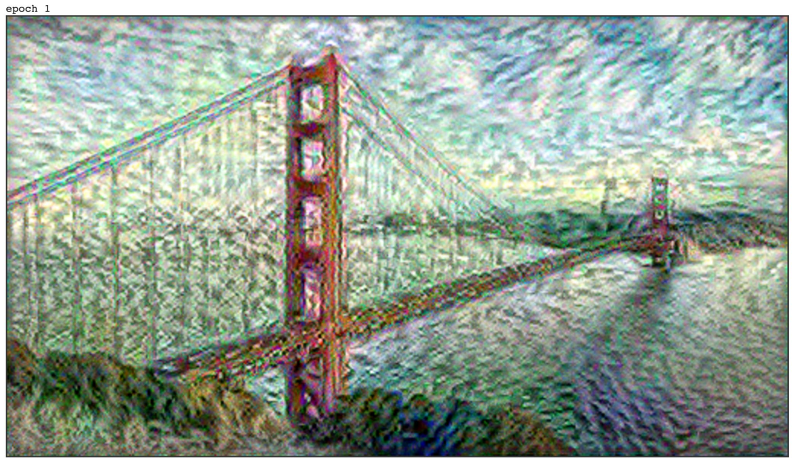


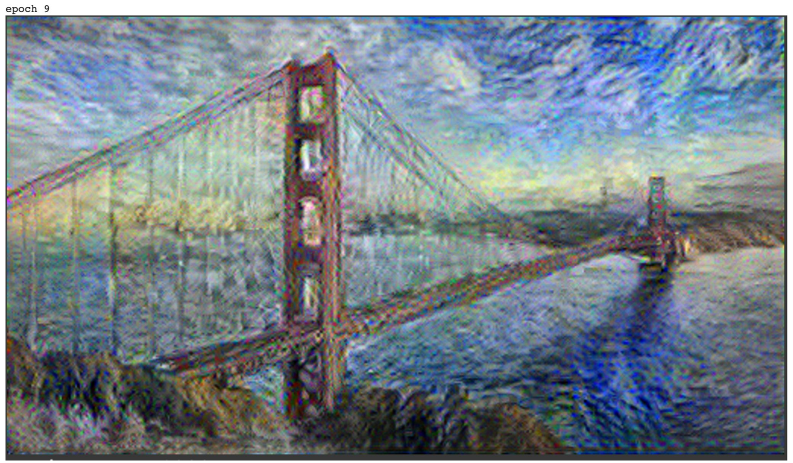
```
block1_conv1
Styles shape: (1, 64, 64)
Styles min: 0.0055228471755981445
Styles max: 28014.55859375
Styles mean: 263.7902526855469
block2_conv1
Styles shape: (1, 128, 128)
Styles min: 0.0
Styles max: 61479.50390625
Styles mean: 9100.9501953125
```

```
block5_conv1
Styles shape: (1, 512, 512)
Styles min: 0.0
Styles max: 110005.3828125
Styles mean: 1487.0380859375
Contents:
block5_conv2
Contents shape: (1, 26, 32, 512)
Contents min: 0.0
Contents max: 2410.879638671875
Contents mean: 13.764151573181152
```

Training Process:

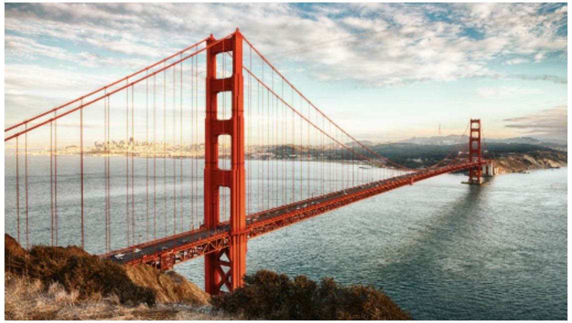
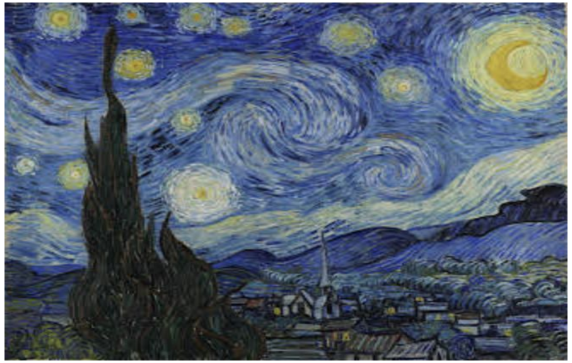
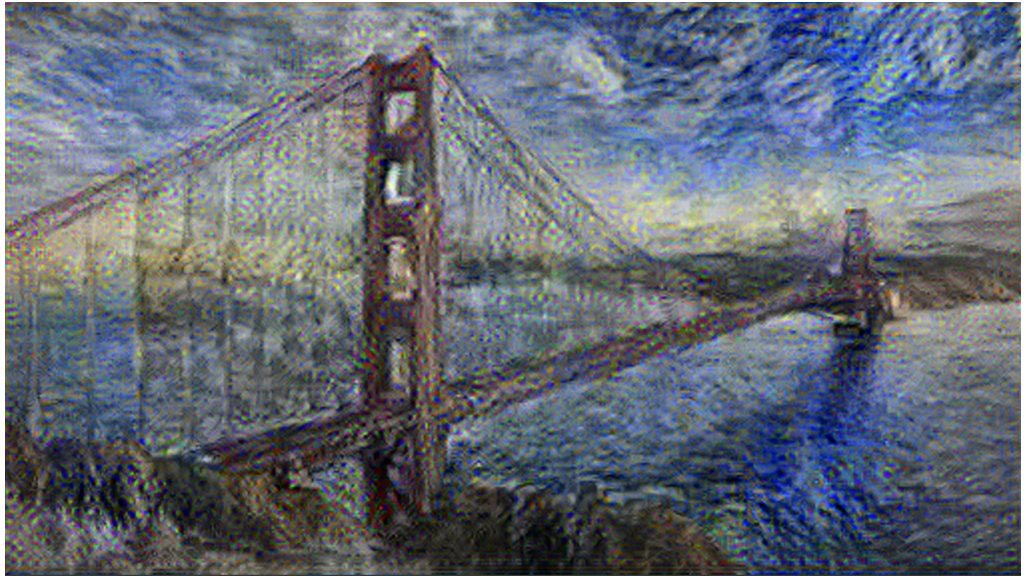
- Loss function: Using this style and content extractor, we can now implement a style transfer algorithm. We do this by calculating the output of each image and the mean square error of the target, and then taking the weighted sum of these loss values. Loss function adds both the style loss and the content loss; then trying to minimize it by gradient descent and update the image.
- We focus on the number of epoch and steps per epoch, because the number of epoch could decide the degree of fusion of images and the number of steps per epoch could decide the sophistication of the style details. Compare epoch 1 with epoch 9:





Results

Above the baseline, our model can train and processing with complicated content image and style image:

Deliverables

Completed Deliverables:

1. Successfully use the convolutional neural network for features selection.
2. Combine a black and white photo is with a cartoon picture to produce a new picture in which the black and white photo is given the color of the cartoon.
3. Improve image spatial resolution and geometric accuracy.
4. When the size of the input image block is set as 16*16, the features of the output image are significant, the contour definition of the original image is high.
5. When the size of the input image block is set as 32*32, the features of the output image are significant, the contour definition of the original image is high.
6. When the size of the input image block is set as 8*8, the features of the output image are significant, the contour definition of the original image is high. (For the image contains too little feature information, which may reduce the accuracy of image classification).

Additional Completed Deliverables:

1. We successfully use the VGG19 CNN model to perfectly transfer art style to generate a styled content image.
2. We could control the degree of art style in the result image by changing the number of epochs and steps per epoch.

What we’ve learned

- This project is related the deep learning lecture that we covered in class, it introduce generative model and give an example: picture of art style transfer. We are inspired by that and decide to finish this project.
- According the set up of our baseline, we just think our project could only give colors to a black white image, but later we are surprisingly finding that we could do real artistic style transfer and produce images that look similar to the great artworks like Dali and Picasso.
- We know how to use existing CNN model to finish our goal.
- As we see, the generated image has noise issues. Thus we still want to implement useful regularization function to reduce noises of the output picture.

Any Questions?

References

[1] *Artistic style transfer with tensorflow*. TensorFlow. (n.d.). Retrieved December 7, 2022, from https://www.tensorflow.org/lite/examples/style_transfer/overview

[2] Simonyan, K., & Zisserman, A. (2015, April 10). *Very deep convolutional networks for large-scale image recognition*. arXiv.org. Retrieved December 7, 2022, from https://arx iv.org/abs/1409.1556

[3] *VGG-19 Convolutional Neural Network.*
https://blog.techcraft.org/vgg-19-convolutional-neural-network/