CV-Final Report

ENGN 6528

Image Stylization via a deep neural network

Group No. 28

Xiao Peng (U6557488)

Pei Liu (U6476079)

Yufei Gu (U6547614)

Shuo Li (U6873158)

04/06/2019

1. Introduction

In the field of image processing, image stylization is a technique for changing the image effect by processing the color, outline, line and other information of an image by computer technology. In recent years, with the development of machine learning, the use of neural networks to deal with image stylization has achieved good results. Based on the deep neural network, based on the research of the existing network, this paper designs two networks in the image stylization of the image processing field to generate images with strong stylization effects.

2. Related work

The global style transforms algorithm processes the image using a spatially invariant transformation method. These methods are very efficient at handling simple transformations such as image segmentation, color or texture allocation and synthesis of Calmes (Yen, et al,2012). For example, A method for image quantization by Amir Semmo is introduced that is based on a dominant color derived from a partial image region (Amir, et al,2016). Chi, MT converts a new image stylization system based on anisotropic reaction diffusion is proposed to facilitate graphics generation and stylized image design. The system begins with a self-organizing pattern created by reaction-diffusion (Chi, et al,2016).

The local style transfer algorithm is based on spatial color mapping and is more efficient and can handle a variety of different categories of applications, such as one-day color transformation, artistic style editing transfer, weather and seasonal changes, and painting stylization. Our work is related to the work of Gatys (Gatys, et al, 2016), using feature mapping of deep-conflicted neural networks with differentiated training, such as the breakthrough performance achieved by VGG-19 in painting style transfer. The main difference with these techniques is that we are the real style transformation of the image. As discussed earlier, the main contradiction is local variation and global consistency. In this respect, our algorithm is related to image processing algorithms. Unlike the techniques of these dedicated scenes, our approach is versatile and can handle a wider variety of image styles.

3. Conceptual design

Image stylization can apply image A’s style to modify image B, then require image C which include A’s style and B’s content. To achieve this function, we need to divide the style and content from one image by using different convolution kernel. Because of the different features of different style, we choose trained VGG19 to do the style and content division.

Layer name	Output size	Layer design (size, # filter)	activation
block1_conv1	32×32	3×3 conv, 64	ReLU
block1_conv2	32×32	3×3 conv, 64	ReLU
block1_pool	16×16	Max 2×2 , stride 2 (“same”)	
block2_conv1	16×16	3×3 conv, 128	ReLU
block2_conv2	16×16	3×3 conv, 128	ReLU
block2_pool	8×8	Max 2×2 , stride 2 (“same”)	
block3_conv1	8×8	3×3 conv, 256	ReLU
block3_conv2	8×8	3×3 conv, 256	ReLU
block3_conv3	8×8	3×3 conv, 256	ReLU
block3_conv4	8×8	3×3 conv, 256	ReLU
block3_pool	4×4	Max 2×2 , stride 2 (“same”)	
block4_conv1	4×4	3×3 conv, 512	ReLU
block4_conv2	4×4	3×3 conv, 512	ReLU
block4_conv3	4×4	3×3 conv, 512	ReLU
block4_conv4	4×4	3×3 conv, 512	ReLU
block4_pool	2×2	Max 2×2 , stride 2 (“same”)	
block5_conv1	2×2	3×3 conv, 512	ReLU
block5_conv2	2×2	3×3 conv, 512	ReLU
block5_conv3	2×2	3×3 conv, 512	ReLU
block5_conv4	2×2	3×3 conv, 512	ReLU
flatten	1×1	2×2×512	
fc_cifa10	1×1	2048×4096	ReLU
		Dropout 0.5	
fc2	1×1	4096×4096	ReLU
		Dropout 0.5	
predictions_cifa10	1×1	4096×10	softmax

Image 1. VGG19

4. Method & algorithms

Our method uses three images, one random initial noise image, one content image, one style image. We plan to use block1_conv1, block2_conv1, block3_conv1, block4_conv1and block5_conv1 to obtain style, as for content, block4_conv1is enough to get a good result. We input the noise image and content image firstly to transfer the content to the noise image and input the noise image and style image secondly to transfer the style to the noise image. But we cannot transfer all styles and contents at one time, so we need to calculate the loss value between noise image and two target images, the do the iteration to get the best results.

5. Equations

F^l: The feature representation of the image at the 1st layer is a matrix with a matrix size of M_l * N_l.
F^l_{ij}: The activation value at position j on the i-th filter of layer 1.
p: original content image
x: generate image
P^l: representation of the original layer in the first layer of CNN
F^l: Generate a representation of the first layer of the image in CNN

$$\mathcal{L}_{\text{content}}(\vec{p}, \vec{x}, l) = \frac{1}{2} \sum_{i,j} \left(F_{ij}^l - P_{ij}^l\right)^2$$

Equation 1. content loss

G^l: Generates a style feature representation of a layer of the image, the size is N_l * N_l

$$G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l.$$

Equation2. The inner product of the i-th and the j-th feature map

A^l: The style feature representation of a layer of a style picture.
M_l: size of the feature map of the first layer
N_l: number of filters in layer 1

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} \left(G_{ij}^l - A_{ij}^l\right)^2$$

Equation 3. style loss of one layer

α: initial style picture

$$\mathcal{L}_{\text{style}}(\vec{a}, \vec{x}) = \sum_{l=0}^{L} w_l E_l$$

Equation 4. total style loss

α: percentage of content
β: percentage of style

$$\mathcal{L}_{\text{total}}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{\text{content}}(\vec{p}, \vec{x}) + \beta \mathcal{L}_{\text{style}}(\vec{a}, \vec{x})$$

Equation 5. total loss

6. Experiment results and conclusions

To fuse the style of the style image with the content of the content image, the generated image content should be as close as possible to the input content image, and the generated image style should be as close as possible to the input style image. Therefore, it is necessary to define a content loss function and a style loss function, which are weighted as a total loss function. The implementation steps are as follows:

- Randomly generate an image
- In each iteration, adjust the pixel value of the image based on the total loss function
- After multiple rounds of iteration, get the optimized picture

The two images are similar in content and cannot be compared simply according to pixels. In order to overcome the problem, we use the ability of CNN for that CNN has the ability to abstract and understand images, so the output of each convolution layer can be considered as the content of images. Like VGG19 which includes multiple convolution layers, pooling layers, and finally the full connection layer

Style is a difficult concept to explain. It can be brushwork, texture, structure, layout, color, etc. in our final project we use the cross-correlation among the features of the convolution layer as the image style. taking conv1_1 as an example, it included 64 feature maps, in other words, it can be understanding as the depth of the image or the number of channels in the image.

For the final result image, we set 5000 iteration to the image and each 100 iteration we can get a processing image. Figure 1 shows the processing from the first image to the result. The first one image is the image with 0.7 noise ratio. And we add the style in the noise, in that case we can retain the content while add the style into the image.

Figure 2 Whole iteration process

Figure 3 Final result image

From figure 2 and 3, we can directly find that the biggest change happens in the first 10 images (1000 iterations), and the more later image, the less differences with the image ahead.

7. observation and discussions

Each feature map is an understanding of the output of the previous layer, which can be liked to 64 different interpretation of the same painting. Different styles lead to different cross-related results. Like the image show below:

Figure 4 different content with different styles

The left column is three content images and the top row five images are style images. From results, we can see the contents after adding different style, but the performance of each style is different that is depending on the style image content.
Secondly, we compared the different ratio between content strength and style strength. Results show follow:

Figure 5 content strength 1

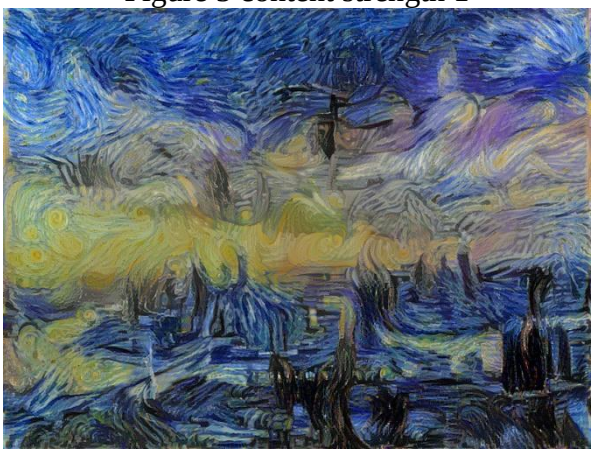

Figure 6 content strength 0.0005

From these two images, we can see some tiny differences. According to the algorithm we can know that total cost comes from content cost and style cost. If the smaller α with the same β , then the total cost would decline, meanwhile the content would become dim.

$$cost\_total = \alpha * cost\_content + \beta * cost\_style$$

So, the smaller content strength, the less content of the original image in the final result covered by more style image.

In the same loss weight, we also discuss the different content layers. Fix the layers of style image and change the layer of content image from $('conv4\_2', 1.)$ to $('conv2\_2', 1.)$. The result shows in figure 7 below:

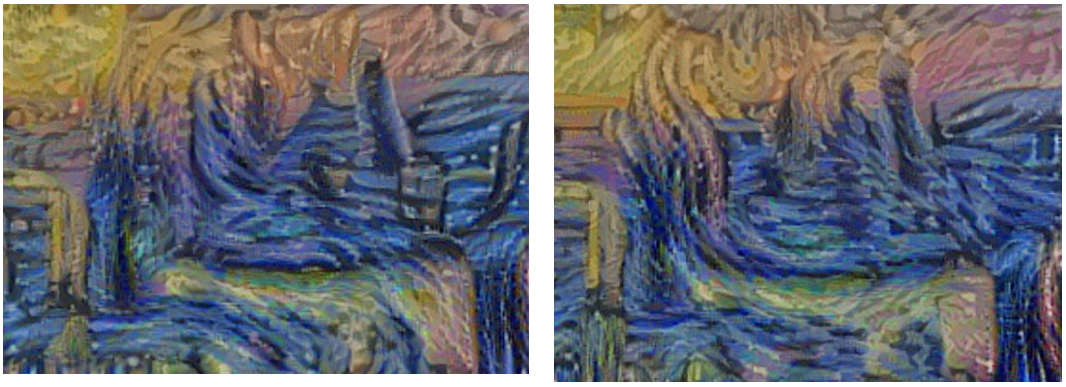

Figure 7 different layers

The right image uses $('conv2\_2', 1.)$ and the left image uses $('conv4\_2', 1.)$. We can see in the picture, the lower the layer of content, the more obvious the content in the resulting image.

Then we test the time of different environment. Depending on our equipment, we choose I7-5700HQ CPU and Nvidia GeForce GTX970M GPU. The speed of GPU is obviously better than CPU and results shows in the table 1 below:

	Time generate each image (100 iteration)
CPU (I7-5700HQ,2.7GHz)	Approx. 7 mins
GPU (Nvidia GeForce GTX970M)	Approx. 30 s

8. Summary of learning outcome

From this project, Our team came to know the knowledge of Deep Learning (DL). Compared to the machine learning (ML), the DL algorithms are more intelligent which can learn by itself to increase the accuracy of prediction. And DL uses the artificial neural network (ANN) which algorithms have many layers of structure like the human biological neural network.

One kind of ANN we used in this project is the convolutional neural network (CNN) which is commonly applied to analysis images and image identification.

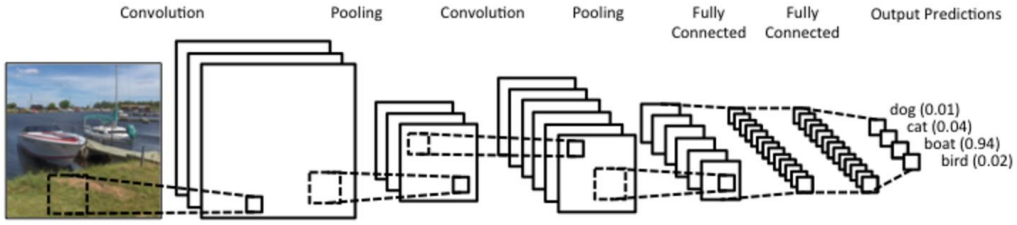

Figure 8 Concept of CNN

The above figure 8 shows the concept of CNN, and we learned three important parts including convolution layer, pooling layer, and fully connected layer.
In this project, we used VGG-19 model (figure 9). VGG-19 has five convolution layers, three fully connected layers and five pooling layers between each layer. In the

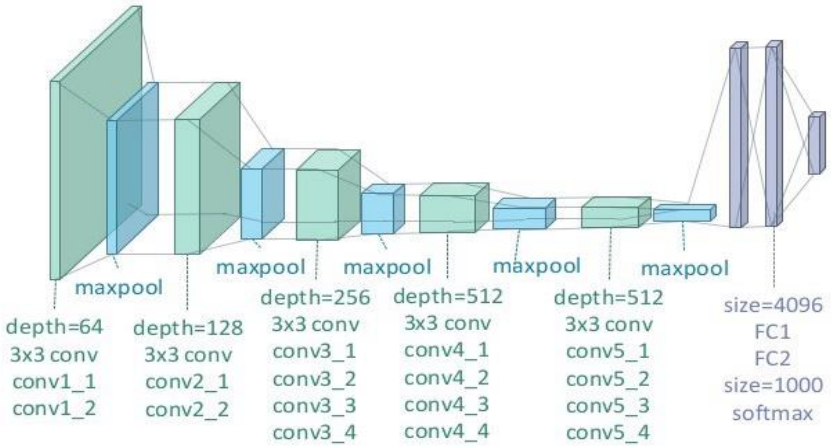

Figure 9 Structure of VGG19

convolution layer, it is using 3x3 convolution kernel matrixes to extract features of images. And pooling layers are aimed to simplify the dimensions without damaging the main features. The purpose of the final connected layer is to classify and output results. Furthermore, we learned how to use python and tensorflow to achieve image stylization.

9. Potential future work

We can increase the proficiency of python, and test different algorithms to improve efficiency and performance of image stylization. And we will continually learn this area and try to build our own CNN model to deal with other problems.

10. Reference

Tairan Z, Congyan L and Junliang X (2018), Realtime Human Segmentation in Video, *Multimedia Modeling*, (206-217). DOI: 10.1007/978-3-030-05716-9_17

Hang Z and Kristin D (2019). Multi-style Generative Network for Real-Time Transfer, *Computer Vision – ECCV 2018 Workshops*, (349-365), DOI: 10.1007/978-3-030-11018-5_32

Yen S. H, Yeh J. P, Lin H. J, Lee T. K＆Pai Y. C (2012), Image stylization, The Imaging Science Journal,60:5,248-255, DOI: 10.1179/1743131X11Y.0000000046

Amir S, Daniel L, Jan E. K, Jürgen D (2016). Image stylization by interactive oil paint filtering, *Computers & Graphics*, Volume 55, 2016, Pages 157-171, ISSN 0097-8493.

Chi M. T, Liu W. C & Hsu S. H (2016). *Vis Comput* 32:1549 https://doi-org.virtual.anu.edu.au/10.1007/s00371-015-1139-2.

Gatys L. A, Ecker A. S, & Bethge M (2016). Image style transfer using convolutional neural networks. *InProceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages2414–2423, 1, 2, 3, 5

Grossfeld B 2017, 'A simple way to understand machine learning vs deep learning', ZenDesk, viewed 4 June 2019, https://www.zendesk.com/blog/machine-learning-and-deep-learning/

Das S 2017, 'CNN Architectures: LeNet, AlexNet, VGG, GoogleLeNet, ResNet and more ...', Medium, viewed 4 June 2019, https://medium.com/@sidereal/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5

11. Teamwork and collaboration

Shuo Li: introduction and related work

Pei Liu: conceptual design, methods, algorithms and equations

Xiao Peng: experiments results, observation and discussion

Yufei Gu: summary, potential future work and readme file

12. Peer assessment

All well done

In the group project this semester, it was the first time we faced the artificial neural network area, but whole team members actively learned the relative knowledge of this project. Everyone always shared the ideas to enhance the understanding of deep learning. As a result, we gradually learned concept and principle of CNN and successfully fished the responsible parts which include code-writing, debug, presentation and report.