DEEP LEARNING LAB(EEE4423-01) Week 10 - Image Style Transfer Using Convolutional Neural Networks, Perceptual Losses for Real-Time Style Transfer and Super-Resolution

Hyeon Si Eun[1], 2019142214
[1] Department of Electric and Electronic Engineering, Yonsei University

1 Image Style Transfer Using Convolutional Neural Networks

"Image Style Transfer Using Convolutional Neural Networks"[1] is a seminal paper by Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge, published in 2016. The paper presents a novel method for combining the content of one image with the style of another image using convolutional neural networks (CNNs). This technique, known as neural style transfer, has since become a popular application in the fields of computer vision and graphics. The authors leverage the power of CNNs, which were already known for their outstanding performance in image recognition tasks, to represent and reconstruct images in a deep feature space. In this space, the content and style of the images can be separated and manipulated independently. For example, the method involves applying the painting style of famous painters to landscape photos. The approach presented in the paper begins with a noise image and reconstructs both the content and style. In this process, the weights of the CNN are fixed, and the noise image is updated. The pre-trained VGG19 CNN model is used in the paper.

The paper demonstrates that their method can successfully transfer various styles to a wide range of content images, producing visually appealing results. Since its publication, neural style transfer has become a popular area of research, with numerous improvements and variations proposed. The technique has also found applications in art, design, and various other creative fields.

The main points of the paper are as follows:

- How is the style of an image defined?
- How can content and style be reconstructed?

Content representation

They define a content loss function to reconstruct the content image. This function is represented by the F matrix, which corresponds to the i^{th} filter present in the l^{th} convolution layer. A layer with N_l distinct filters has N_l feature maps each of size M_l, where M_l is the height times the width of the feature map. So the responses in a layer l can be stored in a matrix $F_l \in R^{N_l \times M_l}$ where F^l_{ij} is the activation of the i^th filter at position j in layer l.

$$L_{content}(\vec{p},\vec{x},l) = \frac{1}{2}\sum_{i,j}(F^l_{ij} - P^l_{ij})^2 \tag{1}$$

By differentiating Equation (1), it can be represented as follows.

$$\frac{\partial L_{content}}{\partial F^l_{ij}} = \begin{cases} (F^l - P^l)_{ij} & if \quad F^l_{ij} > 0 \\ 0 & if \quad F^l_{ij} < 0 \end{cases} \tag{2}$$

p: content image
x: input image (noise)
l: l-th layer
F: Activation value when noise image x is inserted
P: Activation value when content image p is inserted (constant)

In Figure 1, a is a picture reconstructed using a filter that exists in the first layer, and the following is an enlarged picture of a specific part of it. The further to the right, that is, the deeper layer the filter is used, the lower the resolution. This is because filters with deeper layers cover a wider area and express higher-dimensional features. In Content Reconstructions, the initial layers contain detailed pixel information. However, as the layers become deeper, they lose this detailed information.

Style Reconstruction

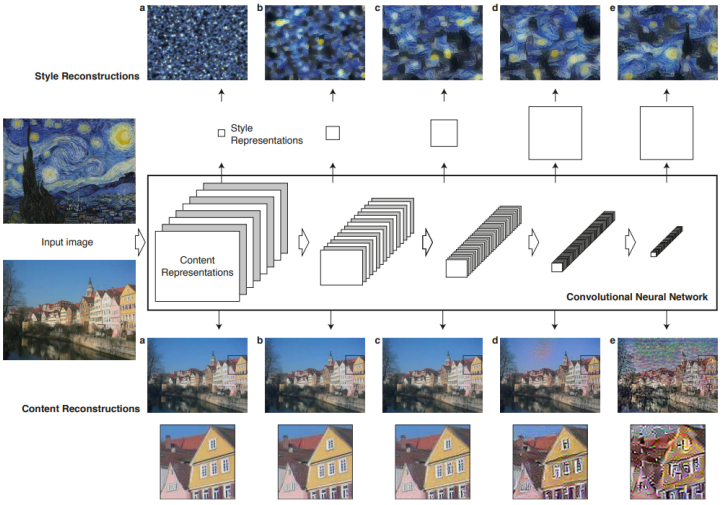

Figure 1: Image representations in a Convolutional Neural Network (CNN). A given input image is represented as a set of filtered images at each processing stage in the CNN.

In Style Reconstructions, correlations between the different filter responses are calculated and used for style reconstruction. That is, 'Style' is defined using a matrix called Gram matrix G. This process has the characteristic of restoring only the style while losing the content of the style image. To get the input image's style representation, we need to use the information in feature space. In this case, Gram matrix $G^l \in R^{N_l \times N_l}$ is used. $G^l_{i,j}$ is the dot product between vectorized feature maps i and j in layer l and is expressed as the following equation (3).

$$G^l_{ij} = \sum_k F^l_{ik} F^l_{jk} \tag{3}$$

The authors define loss in the direction of minimizing the mean squared distance between Gram matrix of the original image and Gram matrix of the generated image.

The content original image is written as $\vec{p}$, but the style original image for learning style is written as $\vec{a}$. The style orignal image's feature map for layer l is denoted by A^l and the generated image's feature map for layer l is l is denoted by F^l.

$$E_l = \frac{1}{4N_l^2 M_l^2}\sum_{i,j}(G^l_{ij} - A^l_{ij})^2 \tag{4}$$

Total style loss is denoted as follows:

$$L_{style}(\vec{a},\vec{x}) = \sum_{l=0}^{L} w_l E_l \tag{5}$$

w_l is weighting factor for each layer for style loss. If Equation (5) is differentiated, it is as follows:

$$\frac{\partial E_l}{\partial F^l_{ij}} = \begin{cases} \frac{1}{N_l^2 M_l^2}((F^l)^T(G^l - A^l))_{ji} & if \quad F^l_{ij} > 0 \\ 0 & if \quad F^l_{ij} < 0 \end{cases} \tag{6}$$

Style Transfer

The authors aim to synthesize an input image $\vec{x}$ using content information from $\vec{p}$ and style information from $\vec{a}$. The final loss function for this purpose is as follows. The final loss function for this is as follows.

$$L_{total}(\vec{p},\vec{a},\vec{x}) = \alpha L_{content}(\vec{p},\vec{x}) + \beta L_{style}(\vec{a},\vec{x}) \tag{7}$$

This is an extended abstract. The full paper is available at the this paper and this paper

α and β represent the weighting factors for content reconstruction and style reconstruction, respectively. In the paper, β is generally larger than α. The authors always resize the size of the style image and content image to be the same.

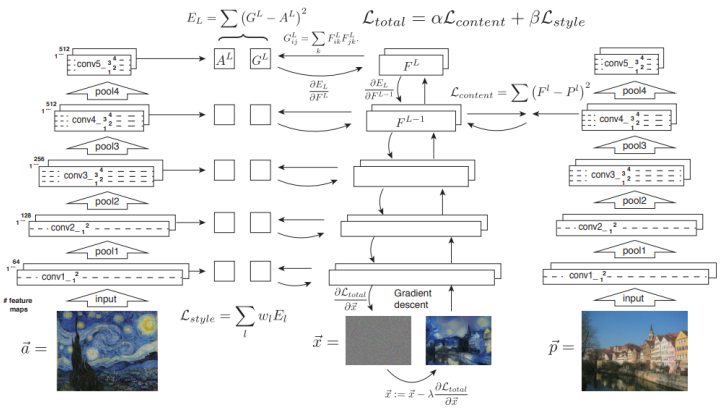

Figure 2: Style transfer algorithm

Experiments

If you want to see detailed experimental results, I recommend viewing the full paper.

2 Perceptual Losses for Real-Time Style Transfer and Super-Resolution

"Perceptual Losses for Real-Time Style Transfer and Super-Resolution"[2] proposes the use of perceptual loss functions for training feed-forward networks for image transformation tasks, instead of using per-pixel loss functions.

1. Per-pixel loss functions

Comparing two images based on their individual pixel values. So, if two images, that are perceptually the same, but different from each other based on even one pixel, then based on per-pixel loss functions they will be very different from each other.

2. Perceptual loss functions

Comparing two images based on high-level representations from pretrained Convolutional Neural Networks (trained on Image Classification tasks, say the ImageNet Dataset).

They evaluate their approach on two image transformation tasks, Style Transfer and Single-Image Super Resolution. For style transfer, they train feed-forward networks that try to solve the optimization problem proposed by Gatys et al. 2015. For super resolution, they experiment with using perceptual losses, and show that it gets better results than using per-pixel loss functions.

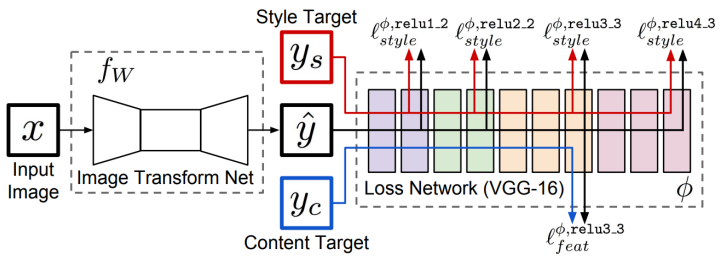

Figure 3: System overview.

The proposed model architecture is composed of two components, Image Transformation Network (f_w) and Loss Network (Φ) The basic principle is to calculate the output image ($\hat{y}$) using the Image Transform Net trained with the original image set. Then, input this output image and the original image together into a pre-trained network model (VGG-16 in this case) to access the hidden layers ($relu1\_2, relu2\_2, relu3\_3, relu4\_3$). This allows for the extraction of feature spaces (maps) and comparison of corresponding features. The reason for this is that the pre-trained image classification model has an appropriate feature map for each image. In other words, one more fixed network that is not trained is added for Loss calculation of Image Transform Net.

These representations are used to define two types of losses:

Feature Reconstruction Loss

Feature reconstruction loss is calculated by extracting two feature spaces of the output image and the original image from the pre-trained model Hidden Layers, and represents the Euclidean distance between the two feature spaces. As for the characteristics of the loss the overall spatial structure of the image is maintained , and other color or texture characteristics are not preserved. For this reason, features can be extracted with specific patterns from approximate edges toward the later part of the pre-trained network model. Therefore, only one output value of the hidden layer at the back end is used to extract features representing the overall structure on the image.

$$l^{\phi,j}_{feat}(\hat{y},y) = \frac{1}{C_j H_j W_j} \left\| \phi(\hat{y})_j - \phi(y)_j \right\|_2^2 \tag{8}$$

ϕ is the feature map of the hidden layer, and j is the depth of the hidden layer. In addition, C, H, and W refer to the channel, height, and width of the calculated feature map, respectively. In this formula, the difference between the output image and the original image feature is represented by the $L2$ norm (Euclidean norm), and normalized by the feature number $(C \times H \times W)$.

Style Reconstruction Loss

Style reconstruction loss is a type of loss that contributes to making the output image express the color or texture of the original image, rather than its overall structure, in contrast to the feature reconstruction loss described above. Since style encompasses various elements such as color, texture, and shape, features are extracted from multiple hidden layers. Euclidean distance is then compared by calculating the dot product of these features with a Gram matrix. Because the dot product contains correlation information between layers, this information can express the complex style of the image. As a result, style reconstruction loss can play a role in altering the structure of an image while preserving the correlation between each layer.

$$G^{\phi}_j(x)_{c,c'} = \frac{1}{C_j H_j W_j} \sum_{h=1}^{H_j} \sum_{w=1}^{W_j} \phi_j(x)_{h,j,c} \phi_j(x)_{h,j,c'} \tag{9}$$

ϕ is the feature map of the hidden layer, and j is the depth of the hidden layer. In addition, C, H, and W refer to the channel, height, and width of the calculated feature map, respectively. It can be obtained by doing the dot product between feature maps in the corresponding layer and normalizing with $(C \times H \times W)$. At this time, the Gram matrix can be calculated by converting the feature map into an array of $C \times (H \times W)$ size, and the size after calculation becomes $C \times C$. Finally, Euclidean distance is calculated to compare the Gram matrices obtained from the feature map between the input image and the original image, and this is defined as Style Loss.

$$l^{\phi,j}_{style}(\hat{y},y) = \left\| \phi(G^{\phi}_j(\hat{y})) - G^{\phi}_j(y) \right\|_F^2 \tag{10}$$

The total loss is typically a weighted sum of the feature reconstruction loss and the style reconstruction loss, in case of style transfer. And just a weighted product of the feature reconstruction loss for the super-resolution. These losses are used to learn the weights of the Image Transformation Network.

Simple Loss Functions

Simple Loss is divided into two types: (1) Pixel Loss and (2) Total Variation Regularization. Pixel Loss can be seen as comparing the pixel Euclidean distance between the output image and the original image, and the formula is as follows.

$$l_{pixel}(\hat{y},y) = \|\hat{y} - \phi(y)\|_2^2 / CHW \tag{11}$$

And Total Variation Regularization plays a role of smoothing the image so that the difference between adjacent values is not large because the continuity of adjacent pixel values in the output image must be preserved.

Experiments

If you want to see detailed experimental results, I recommend viewing the full paper.

[1] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[2] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 694–711. Springer, 2016.