NEURAL STYLE TRANSFER

A Project

Submitted in partial fulfillment for the

Award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE ENGINEERING

Submitted by

(Group-14)

…………………Kunal (18103115)

……Khushansh Jindal (18103119)

……….Aseem Gautam (18103054)

Under the Guidance of

Prof. Sachin Choudhary

Punjab Engineering College (Deemed To Be University), Chandigarh

2020-2021

TABLE OF CONTENT

Contents

MOTIVATION

Painting is a popular form of art. For thousands of years, people have been attracted by the art of painting with the advent of many appealing artworks, e.g., van Gogh's "The Starry Night". In the past, re-drawing an image in a particular style requires a well-trained artist and lots of time. Making computers capable of generating an image in style of another image is a major step in making A.I. creative as it can be related to seeing things in different perspective which would be making it more human like. This scope of further advancements is what motivated us to work on Neural Style Transfer.

Problem

Painting is a popular form of art. For thousands of years, people have been attracted by the art of painting with the advent of many appealing artworks, e.g., van Gogh's "The Starry Night". In the past, re-drawing an image in a particular style requires a well-trained artist and lots of time. Thus far the algorithmic basis of this process is unknown and there exists no artificial system with similar capabilities. Also, Deep neural networks have already surpassed human level performance in tasks such as object recognition and detection. However, deep networks were lagging far behind in tasks like generating artistic artefacts having high perceptual quality until recent times. Creating better quality art using machine learning techniques is imperative for reaching human-like capabilities, as well as opens up a new spectrum of possibilities. And with the advancement of computer hardware as well as the proliferation of deep learning, deep learning is right now being used to create art.

SOLUTION

Style transfer is an important image editing task which enables the creation of new artistic works. Given a pair of examples, i.e., the content and style image, it aims to synthesize an image that preserves some notion of the content but carries characteristics of the style. Here we introduce an artificial system based on a Deep Neural Network that creates artistic images of high perceptual quality. The system uses neural representations to separate and recombine content and style of arbitrary images, providing a neural algorithm for the creation of artistic images. We introduce A Neural Algorithm of Artistic Style that can separate and recombine the image content and style of natural images. The class of Deep Neural Networks that are most powerful in image processing tasks are called Convolutional Neural Networks. Convolutional Neural Networks consist of layers of small computational units that process visual information hierarchically in a feed-forward manner. Each layer of units can be understood as a collection of image filters, each of which extracts a certain feature from the input image. Thus, the output of a given layer consists of so-called feature maps: differently filtered versions of the input image. Our results provide new insights into the deep image representations learned by Convolutional

Neural Networks and demonstrate their potential for high level image synthesis and manipulation.

<u>ARCHITECTURE</u>

Neural style transfer is an optimization technique used to take two images—a *content* image and a *style reference* image (such as an artwork by a famous painter)—and blend them together so the output image looks like the content image, but "painted" in the style of the style reference image.

This is implemented by optimizing the output image to match the content statistics of the content image and the style statistics of the style reference image. These statistics are extracted from the images using a convolutional network. A Convolutional neural network (CNN) is a neural network that has one or more convolutional layers and are used mainly for image processing, classification, segmentation and also for other auto correlated data.

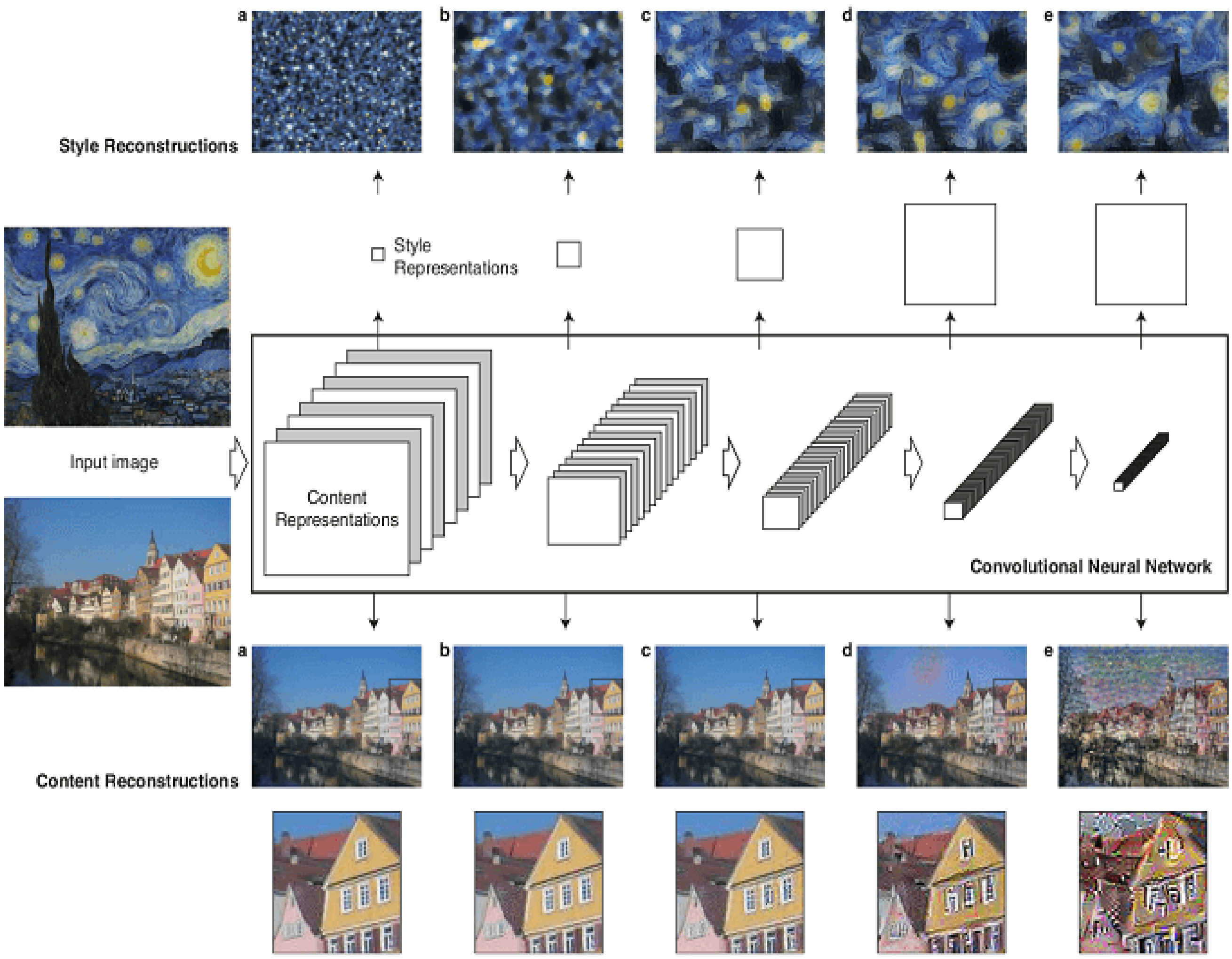
The results were generated on the basis of the VGG network, which was trained to perform object recognition and localization and is described extensively in the original work. We used the feature space provided by a normalized version of the 16 convolutional and 5 pooling layers of the 19-layer VGG network. We normalized the network by scaling the weights such that the mean activation of each convolutional filter over images and positions is equal to one. Such re-scaling can be done for the VGG network without changing its output, because it contains only rectifying linear activation functions and no normalization or pooling over feature maps. We do not use any of the fully connected layers. The model is publicly available and can be explored in the caffe-framework. For image synthesis we found that replacing the maximum pooling operation by average pooling yields slightly more appealing results, which is why the images shown were generated with average pooling

Define content and style representations

Use the intermediate layers of the model to get the *content* and *style* representations of the image. Starting from the network's input layer, the first few layer activations represent low-level features like edges and textures. As you step through the network, the final few layers represent higher-level features—object parts like *wheels* or *eyes*. In this case, you are using the VGG19 network architecture, a pretrained image classification network. These intermediate layers are necessary to define the representation of content and style from the images. For an input image, try to match the corresponding style and content target representations at these intermediate layers.

At a high level, in order for a network to perform image classification (which this network has been trained to do), it must understand the image. This requires taking the raw image as input pixels and building an internal representation that converts the raw image pixels into a complex understanding of the features present within the image. This is also a reason why convolutional neural networks are able to generalize well: they're able to capture the invariances and defining features within classes (e.g. cats vs. dogs) that are agnostic to background noise and other nuisances. Thus, somewhere between where the raw image is fed into the model and the output classification label, the model serves as a complex feature extractor. By accessing intermediate layers of the model, you're able to describe the content and style of input images

BLOCK DIAGRAM



Content Loss

Given a chosen content layer **l**, the content loss is defined as the Mean Squared Error between the feature map **F** of our content image **C** and the feature map **P** of our generated image **Y**.

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

Style Loss

To do this at first we need to, calculate the **Gram-matrix**(a matrix comprising of correlated features) for the tensors output by the style-layers. The Gram-matrix is essentially just a matrix of dot-products for the vectors of the feature activations of a style-layer.
If an entry in the Gram-matrix has a value close to zero then it means the two features in the given layer do not activate simultaneously for the given style-image. And vice versa, if an entry in the Gram-matrix has a large value, then it means the two features do activate simultaneously for the given style-image. We will then try and create a mixed-image that replicates this activation pattern of the style-image. If the feature map is a matrix **F**, then each entry in the Gram matrix **G** can be given by:

$$G_{ij} = \sum_{k} F_{ik}F_{jk}$$

The loss function for style is quite similar to out content loss, except that we calculate the Mean Squared Error for the Gram-matrices instead of the raw tensor-outputs from the layers.

$$\mathcal{L}_{style} = \frac{1}{2}\sum_{l=0}^{L}(G_{ij}^{l} - A_{ij}^{l})^2$$

Total Loss

We jointly minimize the distance of the feature representations of a white noise image from the content representation of the photograph in one layer and the style representation of the painting defined on a number of layers of the Convolutional Neural Network. The loss function we minimize

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

where α and β are the weighting factors for content and style reconstruction, respectively.

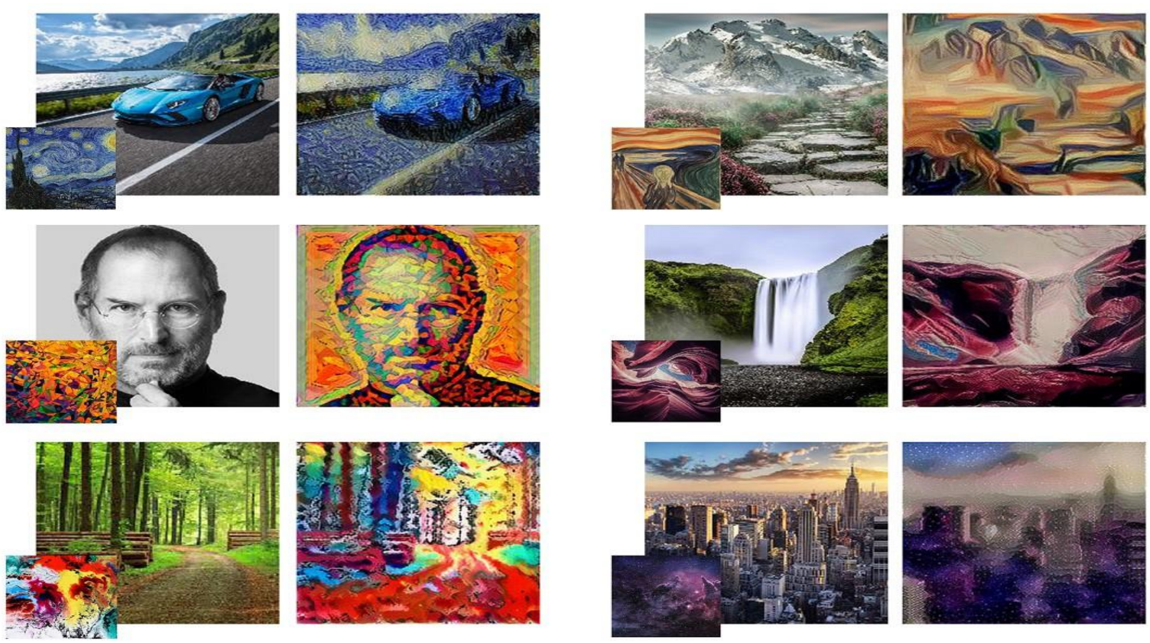
LANGUAGES

- Python
- CNN Algorithm
- TensorFlow
- VGG19

DATASET

ImageNet- **VGG19** is a model, with weights pre-trained on **ImageNet**, is a dataset of over 15 millions labeled high-resolution images with around 22,000 categories.

OUTPUT



APPLICATIONS

- Photos and Video editors
- Commercial art
- Gaming
- Virtual Reality

CONCLUSION

The key finding of this project is that the representations of content and style in the Convolutional Neural Network are well separable. That is, we can manipulate both representations independently to produce new, perceptually meaningful images. It could be improved through increasing the number of iteration, or by trying out a different style transfer algorithm which could preserve the edges of the base image, or by trying out with different optimizer to minimize gradient and loss.

REFERENCES

1. Leon A. Gatys, Alexander S. Ecker, Matthias Bethge: *Image Style Transfer Using Convolutional Neural Networks, CVPR 2016*.
2. Yijun Li ,Chen Fang ,Jimei Yang ,Zhaowen Wang ,Xin Lu ,Ming-Hsuan Yang*: Universal Style Transfer via Feature Transforms, NIPS 2017.*
3. Leon A. Gatys, Alexander S. Ecker, Matthias Bethge: *A Neural Algorithm of Artistic Style,CVPR 2015.*
4. Yongcheng Jing, Zunlei Feng, Jingwen Ye, Mingli Song: *Neural Style Transfer: A Review,CVPR 2018.*