Deep Photo Style Transfer

Darshan Jethwa
Pavan Soundara
Tarun Mogra
Vatsal Thakar
Table of content

- Introduction
- Previous work
- Challenges
- Method
- Comparisons
- Conclusion & Future Work
Introduction

- Transfer the style of a reference style photo onto another input picture.
Why deep photo style transfer?

- Artistic image transformation
- Altering the time of day or weather of a picture
Previous Work

- Use deep learning to extract the features of the style and the input image
- Limited in the diversity of scenes or transfers
Content Extractor

Deep convolutional neural network VGG 19

Input layer  Hidden layer  Output layer

winter wolf image

Content of winter wolf image
Style Extractor

Deep convolutional neural network VGG 19

Input layer

Hidden layer

Output layer

The Scream painting

Content of The Scream painting

Texture of The Scream painting

Texture estimator, based on Gram matrix correlations
Merger

- Done with the help of an optimization problem.
- We define a cost function, which we want to minimize.
- Cost function decreases in every iteration.
- We use Gatys et al. approach to create a cost function which penalizes the synthesized image if its content was not equal to the desired content and its style was not equal to the desired style.
General Steps

- Synthesize a noise image
- Extract the content and the style of our_image
- Calculate the distance between the content of our_image and the content of content_image
- Calculate the distance between the style of our_image and the style of style_image
- Calculate the cost function and the gradient
- Try to minimise the cost function and set threshold to stop the cost function
Background

- Transfers the reference style image $S$ onto the input image $I$ to produce an output image $O$ by minimizing the objective function.

$$
\mathcal{L}_{\text{total}} = \sum_{\ell=1}^{L} \alpha_{\ell} \mathcal{L}_{c}^{\ell} + \Gamma \sum_{\ell=1}^{L} \beta_{\ell} \mathcal{L}_{s}^{\ell}
$$

with:

$$
\mathcal{L}_{c}^{\ell} = \frac{1}{2N_{\ell}D_{\ell}} \sum_{ij} (F_{\ell}[O] - F_{\ell}[I])_{ij}^2
$$

$$
\mathcal{L}_{s}^{\ell} = \frac{1}{2N_{\ell}^2} \sum_{ij} (G_{\ell}[O] - G_{\ell}[S])_{ij}^2
$$

- $L$ is the total number of convolutional
- $D$ is the vectorized feature map
- $N$ is the number of filter in each layer
- $\alpha$ and $\beta$ are the weights to configure layer preferences
- $\Gamma$ is a weight that balances the tradeoff between the content
- $F$ is feature matrix
- $G$ is gram matrix
Challenges

- Structure preservation
- Semantic Accuracy and Transfer Faithfulness.
New Solution

- Deep-learning approach to photographic style transfer
- Photorealism regularization
- Augmented style loss with semantic segmentation
Regularization

- Optimization scheme to preserve the structure
- Affine function that maps the input RGB values onto their output counterparts
- Apply regularization on the transformed image rather than final output image
- We name $V_c[O]$ the vectorized version $(N \times 1)$ of the output image $O$ in channel $c$ and define the following regularization term that penalizes outputs that are not well explained by a locally affine transform:

$$\mathcal{L}_m = \sum_{c=1}^{3} V_c[O]^T M_I V_c[O]$$

- $M_I$ is the standard linear system
Augmented style loss with semantic segmentation

- Gram matrix calculated over the whole image
- It can cause “spillovers”
- To overcome we create a mask for same structures as clouds, buildings etc.
- Then add this mask to input image as channels
Augmented style loss with semantic segmentation

\[
L^t_s = \sum_{c=1}^{C} \frac{1}{2N^2_{\ell,c}} \sum_{ij} (G_{\ell,c}[O] - G_{\ell,c}[S])_{ij}^2
\]

\[
F_{\ell,c}[O] = F_{\ell}[O]M_{\ell,c}[I] \quad F_{\ell,c}[S] = F_{\ell}[S]M_{\ell,c}[S]
\]

where \(C\) is the number of channels in the semantic segmentation mask, \(M_{\ell,c}[\cdot]\) denotes the channel \(c\) of the segmentation mask in layer \(\ell\), and \(G_{\ell,c}[\cdot]\) is the Gram matrix corresponding to \(F_{\ell,c}[\cdot]\). We downsample the masks to match the feature map spatial size at each layer of the convolutional neural network.
New approach

\[ \mathcal{L}_{\text{total}} = \sum_{l=1}^{L} \alpha_{l} \mathcal{L}_{c}^{l} + \Gamma \sum_{\ell=1}^{L} \beta_{\ell} \mathcal{L}_{s}^{\ell} + \lambda \mathcal{L}_{m} \]

- \( \lambda \) is the weight that controls the photorealism regularization
- \( \mathcal{L}_{c} \) is same as the previous solution
- \( \mathcal{L}_{s} \) computed recently
New approach
Implementation

(1): nn.TVLoss
(2): cudnn.SpatialConvolution(3 -> 64, 3x3, 1,1, 1,1)
(3): cudnn.ReLU
(4): nn.StyleLoss
(5): cudnn.SpatialConvolution(64 -> 64, 3x3, 1,1, 1,1)
(6): cudnn.ReLU
(7): cudnn.SpatialMaxPooling(2x2, 2,2)
(8): cudnn.SpatialConvolution(64 -> 128, 3x3, 1,1, 1,1)
(9): cudnn.ReLU
(10): nn.StyleLoss
(11): cudnn.SpatialConvolution(128 -> 128, 3x3, 1,1, 1,1)
(12): cudnn.ReLU
(13): cudnn.SpatialMaxPooling(2x2, 2,2)
(14): cudnn.SpatialConvolution(128 -> 256, 3x3, 1,1, 1,1)
(15): cudnn.ReLU
(16): nn.StyleLoss
(17): cudnn.SpatialConvolution(256 -> 256, 3x3, 1,1, 1,1)
(18): cudnn.ReLU
(19): cudnn.SpatialConvolution(256 -> 256, 3x3, 1,1, 1,1)
(20): cudnn.ReLU

(21): cudnn.SpatialConvolution(256 -> 256, 3x3, 1,1, 1,1)
(22): cudnn.ReLU
(23): cudnn.SpatialMaxPooling(2x2, 2,2)
(24): cudnn.SpatialConvolution(256 -> 512, 3x3, 1,1, 1,1)
(25): cudnn.ReLU
(26): nn.StyleLoss
(27): cudnn.SpatialConvolution(512 -> 512, 3x3, 1,1, 1,1)
(28): cudnn.ReLU
(29): nn.ContentLoss
(30): cudnn.SpatialConvolution(512 -> 512, 3x3, 1,1, 1,1)
(31): cudnn.ReLU
(32): cudnn.SpatialConvolution(512 -> 512, 3x3, 1,1, 1,1)
(33): cudnn.ReLU
(34): cudnn.SpatialMaxPooling(2x2, 2,2)
(35): cudnn.SpatialConvolution(512 -> 512, 3x3, 1,1, 1,1)
(36): cudnn.ReLU

```
Iteration 50 / 2000
Content 1 loss: 916780.468750
Style 1 loss: 17581.158376
Style 2 loss: 196835.701587
Style 3 loss: 35165.081884
Style 4 loss: 1087201.154094
Style 5 loss: 191.424818
Total loss: 2253754.989509
```
Our Results
Our Results
Comparison and results
Conclusion & Future Works

- Implement the process in the live applications.
References

- https://pdfs.semanticscholar.org/7568/d13a82f7afa4be79f09c295940e48ec6db89.pdf
Questions ?