Style Transfer: Implementation of the method and dash-app

Omkar Mehta (omehta2), Anurag Anand (anuraga2)

March 2022

1 Problem Statement

For this project we will be working on the problem of texture transfer. The primary goal of this exercise is to create a web application where the user can upload an image. This image is an input to the style transfer and texture transfer procedure so that it reflects the semantic content of the input image and also preserves the style from many different styles, available to the user.

We propose using the style transfer method in the ”Image Style Transfer Using Convolutional Neural Nets” paper [1] in PyTorch, which uses pre-trained VGG19 [2] model from Simonyan and Zisserman, 2014 paper, trained on ImageNet data. Extending this, we wish to know the effects of different models on the style transfer and let the users see the difference, for which we will train our own model on Tiny ImageNet dataset, and compare the results of the style transfer, given by different models.

2 Deliverables

1. We will deploy a Dash and Flask-based web app, using Fast Dash [3] that allows users to upload or click a photo, choose a style from many different well-known styles, choose a model of their own choice (vgg-16 [2], resnet18 [4] or our trained model) that produces a new image that reflects the content of the photo and the artistic style of the other.

2. Trained Model: we will design a CNN model architecture and train it on Tiny ImageNet dataset [5]. We intend to design the model architecture in such a way that it aggressively downsamples the input and uses the ideas from GoogleNet's Inception module, ResNeXt's module and Global average pooling to get us a feature map of the input.

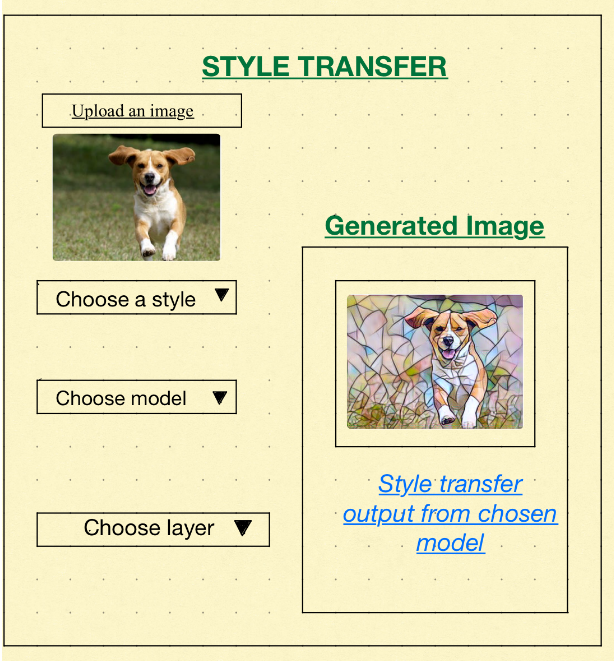
3 Methodology

1. The Image style transfer paper lists down two losses for training the networks. These losses are content loss ($\mathcal{L}_{\text{content}}$) and style loss ($\mathcal{L}_{\text{style}}$). The total loss is computed using the equation $\mathcal{L}_{\text{total}} = \alpha\mathcal{L}_{\text{content}} + \beta\mathcal{L}_{\text{style}}$, where α and β are the weighting factors for content and style construction, respectively. In our application we will be implementing this total loss.

2. Additionally, we will be implementing the idea of perceptual loss from this paper [6] for smoothing out the output image and getting a super resolution, called as spatial smoothness.

3. In the deployed web app, the user will have the option to choose the model from a drop-down menu. Once the model is chosen, the user can choose the MLP layer from which the feature map of the input image is considered. The user will then choose the style from many well-known styles, listed in the drop-down menu.

4. Fast Dash module build the web interfaces without the hassle of developing UI from scratch. We will use Google Cloud Run to deploy the containerized application and host it via Fast Dash. The layout is in Figure 1.



(a) Layout

Figure 1: Web App of Style Transfer

4 Relationship to team members' background

Omkar is pursuing MS in Industrial Engineering (non-thesis). Anurag is pursuing MS in Statistics. We have never worked on applications of computer vision, before.

References

[1] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2414–2423.

[2] "Vgg-16 pre-trained model from pytorch," https://pytorch.org/hub/pytorch_vision_vgg/.

[3] "Fast dash: Open source, python-based tool to develop web applications lightning fast," https://dkedar7.github.io/fast_dash/.

[4] "Resnet-18 pre-trained model from pytorch," https://pytorch.org/hub/pytorch_vision_resnet/.

[5] “Dataloader for tinyimagenet dataset,” https://github.com/pranavphoenix/TinyImageNetLoader.

[6] F.-F. L. Johnson J., Alahi A., “Perceptual losses for real-time style transfer and super-resolution,” in *Computer Vision – ECCV 2016*, vol. 9906, 2016, p. Lecture Notes in Computer Science.