Neural Style Transfer: A Tech Report

Jitesh Jain[†] 19114039 **Gagan Sharma**[†] 19114032 **Divyansh Agarwal**[†] 19115055

R Chinmay[†] 19114067 **Md Junaid Mahmood**[†] 19116040

Group **1**
Indian Institute of Technology Roorkee[*]

Abstract

Neural Style Transfer is one of the earliest application of *deep learning* in the field of creative art. However, the existing technique has not changed since the first method which was proposed in 2015. Since then, there have been many recent developments in the field of *deep learning* and *creative art* including the formulation of new optimizers, losses and schedulers. In this proposal, we present our plan for working on a tech report where we study various factors responsible for obtaining the optimal result.

1 Introduction

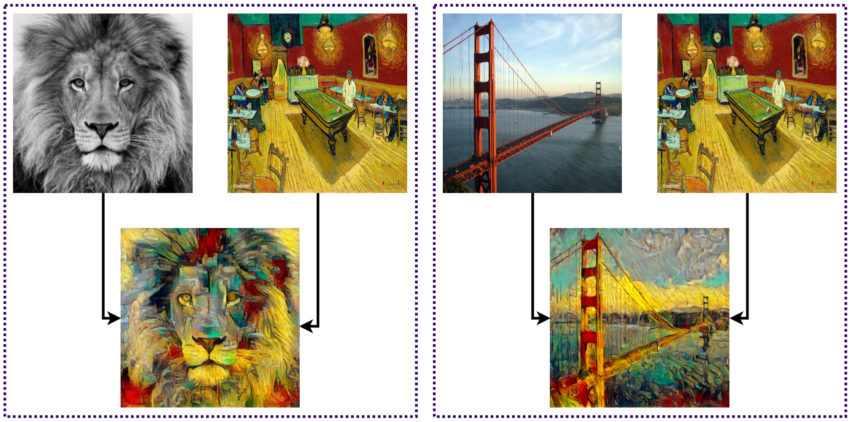

Figure 1: **Neural Style Transfer** We learn a stylized image representation that contains structures from the *content image* and textures from the *style image*.

Neural Style Transfer (NST) was first proposed by Gatys et al. in [2]. Since then, the stylizing images using filters for creative art and image editing have gained popularity among the general public. In this proposal, we first briefly describe the existing method in Sec. 3 and then list the experiments in Sec. 4 that we plan to include in our report.

[*] All authors contributed equally to the project. All authors will work under the supervision of Professor Pravendra Singh at the Department of Computer Science and Engineering, IIT Roorkee.

Proposal for Course Project under CSN-526: Machine Learning.

2 Related Works

A Neural Algorithm of Artistic Style. Gatys et al. first proposed the *Neural Style Transfer* algorithm in [2] in 2015. They formulated style transfer as an optimization problem. They proposed to first extract content and style features from corresponding images and then optimize a learnable image using the combination of features. The result is a stylized image with guidance from the content and style features as shown in Fig. 1. They didn't use any dataset but only a content and style image during training which makes this algorithm widely accepted.

Arbitrary Style Transfer in Real-Time. Huang et al. [5] proposed the use of Adaptive Instance Normalization (AdaIN) to combine content from the content image and style from the style image by transferring feature statistics using an encoder-decoder structure. This work gives impressive style transfer performance.

In our work, we plan to use the algorithm proposed in [2] for our experiments and ablation studies as described in Sec. 3. Owing to the impressive performance of [5], we plan to use the stylized images produced using their pretrained models as pseudo-ground truths as described in Sec. 4.

3 Method

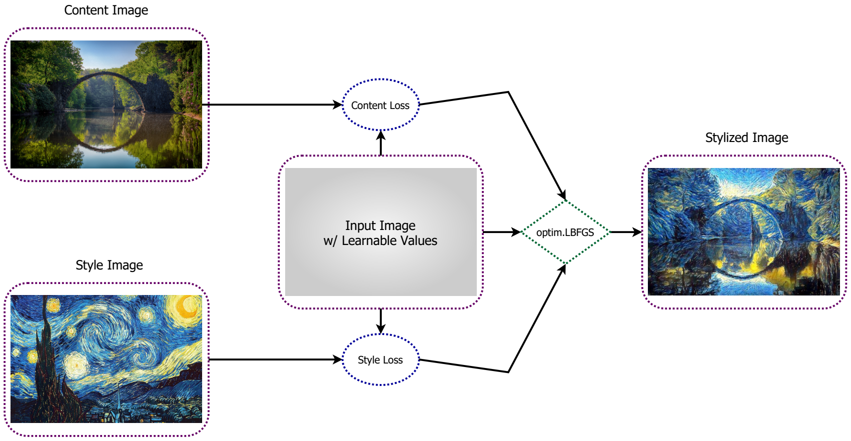

Figure 2: **Neural Style Transfer Pipeline** NST learns the *Input Image* with supervision from the *Content Image* and *Style Image*.

Our method is mainly inspired by [2]. We describe the baseline method briefly and then mention the changes that we plan to try.

Neural Style Transfer is an optimization problem, which takes in 3 images as input: *content image*, *style image* and a *learnable input image* as shown in Fig. 2. The goal is to learn the input image such that the final result consists of the content of the Content image and style of the Style image. We extract the content and style (*gram matrices*) features from any given image by using a deep neural network (VGG19 [12] in [2]). The shallow layers in feature extractor store pixel colour information, and style information and the deeper layers store content (structure) related information. We learn the input image with supervision from a weighted sum of content loss: between the features from content and input image, and style loss: between the features from style and input image as shown in Eq. (1).

$$\mathcal{L}_{total} = \lambda_{content}\mathcal{L}_{content} + \lambda_{style}\mathcal{L}_{style} \tag{1}$$

$\lambda_{content}$ and λ_{style} are hyperparameters. In addition to these, we also assign weights to level wise feature differences for both style and content losses. [2] used LBFGS as their optimizer. In the following section (Sec. 4), we lay out the plan for our proposed experiments.

4 Experiments

Implementation. We will use PyTorch [11] for developing our implementation. Since Neural Style Transfer only requires two images (*style* and *content*) during training, we will not be using any specific dataset. Instead, we will collect a few images off the web. We will conduct our experiments on the Google Colab platform.

Metrics. If possible, we also plan to compare and evaluate all our models based on the *mean-squared error* (MSE) and *structural similarity index measure* (SSIM) scores with respect to *pseudo-ground truths* generated using the method proposed in [5].

4.1 Feature Extractors

Gatys et al. used the VGG [12] network as their feature extractor in [2]. Recently, there have been many less bulky and faster recently proposed models like ResNet [3] and DenseNet [4]. We plan to conduct an ablation study using ResNet and other relevant models available on PyTorch Model Zoo. If time permits, we can also experiment with vision transformers [1].

4.2 Optimizers

The model's performance and convergence time during training depend majorly on the optimizer's choice. [2] used LBFGS in their original work. We plan to check the effect of using Adam [7] and AdamW [9] on the performance.

4.3 Loss

Gatys et al. [2] used a weighted sum of *content loss* ($\mathcal{L}_{content}$) and *style loss* ($\mathcal{L}_{style}$) for the original criterion. We plan to use a weighted total-variation [13] loss ($\mathcal{L}_{tv}$) for performance improvement. Along with these 3 losses, we plan to conduct ablation on the usage of a few more losses: laplacian loss [8] ($\mathcal{L}_{lap}$) and perceptual loss [6] ($\mathcal{L}_{pl}$) as shown in Eq. (2) with λ_i being tunable hyperparameters.

$$\mathcal{L}_{total} = \lambda_{content}\mathcal{L}_{content} + \lambda_{style}\mathcal{L}_{style} + \lambda_{tv}\mathcal{L}_{style} + \lambda_{pl}\mathcal{L}_{pl} \tag{2}$$

If time permits, we may also try matching style features in a patch-wise manner inspired by Swapping-Autoencoder [10]. In addition to this, we can also try using a reconstruction loss treating the image generations with [5] as pseudo target images.

4.4 Number of Iterations

The *number of iterations* (N) is a critical hyperparameter that determines the tradeoff between performance and training time for neural style transfer. We plan to plot the effect of changing the number of iterations to our performance on the *MSE* and *SSIM* metrics. We are hopeful that we will establish a pattern and formulate the tradeoff.

5 Conclusion

We presented our plan to work on conducting a technical study for the Neural Style Transfer Task in this proposal as a part of the **CSN-526: Machine Learning Course Project**. At the end of the project, we plan to release our tech report on arXiv.

6 Acknowledgment

We thank Professor Pravendra Singh at the Department of Computer Science and Engineering, IIT Roorkee for providing us with an opportunity to work on the project.

References

[1] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 3

[2] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv*, 2015. 1, 2, 3

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016*, 2016. 3

[4] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 3

[5] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017. 2, 3

[6] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. *arXiv*, 2016. 3

[7] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 3

[8] Shaohua Li, Xinxing Xu, Liqiang Nie, and Tat-Seng Chua. Laplacian-steered neural style transfer. In *ACMMM*, 2017. 3

[9] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 3

[10] Taesung Park, Jun-Yan Zhu, Oliver Wang, Jingwan Lu, Eli Shechtman, Alexei A. Efros, and Richard Zhang. Swapping autoencoder for deep image manipulation. In *Advances in Neural Information Processing Systems*, 2020. 3

[11] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 3

[12] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 2, 3

[13] Wikipedia. Total variation. 3