Neural Style Transfer: Mid-Term Report

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

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

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

Figure 1: **Neural Style Transfer Results** We are able to reproduce the baseline and achieve competitive performance to the pseudo GTs. We trained for 1000 iterations with ResNet-18 [2] feature extractor and LBFGS optimizer for the above results.

Abstract

We continue our work on ablating several factors responsible for achieving good performance on the neural style transfer task as we mentioned in our proposal. In this mid-term report, we present our experimental settings and findings when ablating on the feature extractor, optimizer, and the number of iterations. We propose to use pseudo ground truths to evaluate the performance of our methods. We find that ResNet-18 and LBFGS show the best performance for feature extractor and optimizer, respectively. Our code is publicly available at https://github.com/praeclarumjj3/NST-Tech.

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

Mid-Term Report for Course Project under CSN-526: Machine Learning.

1 Introduction

Neural Style Transfer (NST) was first proposed by Gatys et al. in [1]. Since then, the stylizing images using filters for creative art and image editing have gained popularity among the general public. However, to the best of our knowledge, there is no existing technical study that ablates the several factors like feature-extractors, optimizers, number of iterations, initialization of the input image, and the loss terms. We work on developing an extensive study conducting ablation studies with different settings. In this mid-term report, we present our progress and findings so far in the project. Since there exists no real ground truth for the stylized image, we propose to use the results from the published work by Huang *et al.* [3] as pseudo ground truths for quantitatively evaluating our results.

We first briefly explain our method in Sec. 2. We then present our experimental settings and results in Sec. 3. Lastly, we present the future plan of our project in Sec. 4.

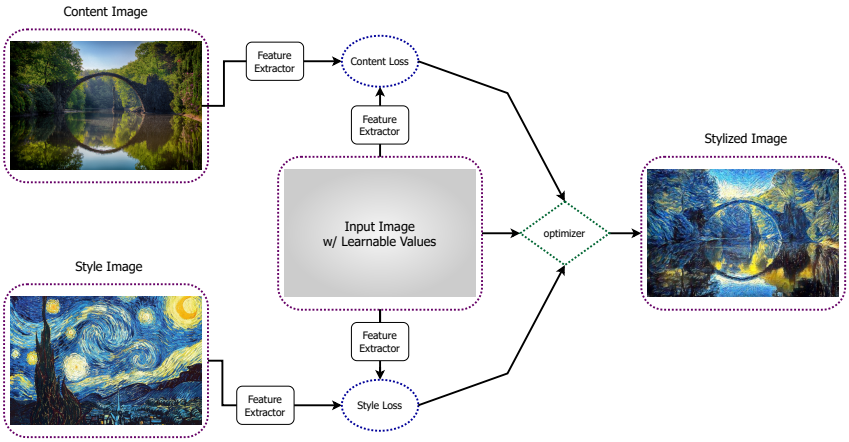

Figure 2: **Neural Style Transfer Pipeline** NST learns the *Input Image* with supervision from the *Content Image* and *Style Image*.

2 Method

Our method is mainly inspired by [1]. 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 (like VGG16 or VGG19 or ResNet-18). 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. We set $\lambda_{content} = 1 \times e^0$ and $\lambda_{style} = 1 \times e^6$ for all our experiments.

3 Experiments

3.1 Data and Metrics

Data. For training and evaluating our methods, we choose 5 pairs of content and style images from the web as shown in Fig. 3. We average the metric scores over the 5 pairs for recording the final scores.

Metrics. Since there exists no real ground truth for the stylized image, we use the results from the Adaptive Instance Normalization based method proposed by Huang *et al.* [3] as pseudo ground truths (Fig. 3) for quantitatively evaluating our results. We use the Root Mean Square Error (RMSE), Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) metrics for comparing our results with the pseudo ground-truths.

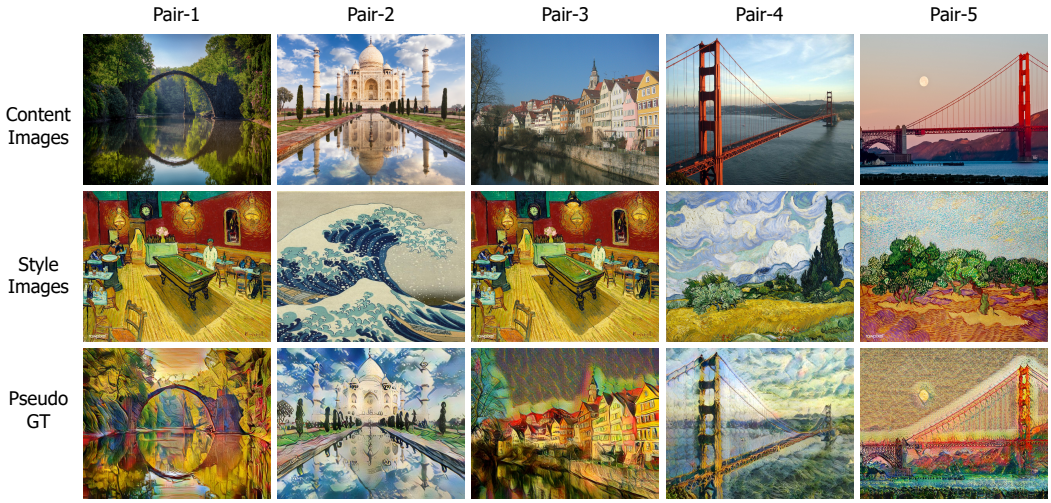

Figure 3: **Training Data Pairs.** We use 5 pairs of content and style images taken from the web for all our experiments.

3.2 Implementation Details

We implement our method in PyTorch [7]. We use the publicly available models on the torchvision model zoo for our feature extractors. When using VGG-16 [8] as our feature extractor, we use the features from the `relu2_2` layer for calculating the content loss and the gram matrices from [`relu1_2`, `relu2_2`, `relu3_3`, `relu4_3`] layers for calculating the style loss. In case of VGG-19 [8], we use `relu4_2` layer for content loss and [`relu1_1`, `relu2_1`, `relu3_1`, `relu4_1`, `relu4_2`, `relu5_1`] for style loss. When using ResNet-18 [2], we use features from `layer1` for content loss and the gram matrices from [`conv1`, `layer1`, `layer2`, `layer3`] for style loss.

Unless stated otherwise, we train for 1000 iterations with the LBFGS optimizer with the Resnet-18 feature extractor.

3.3 Ablation on Feature Extractors

We experiment with three different feature extractors: VGG-16 [8], VGG-19 [8] and ResNet-18 [2]. We compare the three models on RMSE, SSIM, and PSNR values. We observe that ResNet-18 gives the best performance as shown in Tab. 1. Thus, we use ResNet-18 as our feature extractor for further experiments.

3.4 Optimizers

We experiment with different optimizers among LBFGS, Adam [4] and AdamW [6] using the ResNet-18 extractor. In order to find the best learning rate (lr) in case of Adam and AdamW, we tune the lr hyper-parameter on Pair-2 (Fig. 3). We find lr $= 1e^{-1}$ and lr $= 1e^{-2}$ to be the best setting for Adam

Metric	VGG-16	VGG-19	ResNet-18
RMSE ↓	0.01213	0.01231	**0.01187**
PSNR ↑	38.336	38.211	**38.528**
SSIM ↑	0.8798	0.8767	**0.8864**

Table 1: **Ablation on Feature Extractors.** ResNet-18 gives the best scores on RMSE, SSIM and PSNR.

and AdamW, respectively. Evaluation for tuning the learning rate was also done using the PSNR, SSIM and RMSE metrics. We find LBFGS to show the best performance and hence, use LBFGS for further experiments as shown in Tab. 2.

Metric	LBFGS	Adam	AdamW
RMSE ↓	**0.01187**	0.01270	0.01238
PSNR ↑	**38.528**	38.001	38.235
SSIM ↑	**0.8864**	0.8785	0.8801

Table 2: **Ablation on Optimizers.** LBFGS gives the best scores on RMSE, SSIM and PSNR.

3.5 Number of Iterations

We also experiment with the number of iterations for the ResNet-18 feature extractor and LBFGS optimizer. We selected three numbers 1000, 1500, 2000 and used the same methodology in order to compare and evaluate the three models using different numbers of iterations. We find that the 1000 iterations gives the best performance as shown in Tab. 3. Thus, we set the number of iterations to 1000 for further experiments.

Metric	1000	1500	2000
RMSE ↓	**0.01187**	0.01191	0.01191
PSNR ↑	**38.528**	38.504	38.501
SSIM ↑	**0.8864**	0.8859	0.8860

Table 3: **Ablation on Number of Iterations.** 1000 iterations give the best scores on RMSE, SSIM and PSNR.

4 Future Work

We plan to perform more ablation experiments to study the effects of initialization techniques of the input image, initializing it with content image (default), or style image or any random image. We also plan to include tv-loss and laplacian-loss [5] (if time permits) to the total loss term. We may also try more feature extractors like ResNet-34 [2], if time permits.

We plan to complete the initialization and loss experiments by the end of March. We will then work on writing the report and releasing it on arXiv by Mid-April.

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

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016*, 2016. 1, 3, 4

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

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

[5] Shaohua Li, Xinxing Xu, Liqiang Nie, and Tat-Seng Chua. Laplacian-steered neural style transfer. In *ACMMM*, 2017. 4

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

[7] 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

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