CS 484 Introduction to Computer Vision
Project Proposal
A Generative Style Transfer Model

Boran Kılıç - 22103444
Defne Yaz Kılıç - 22102167
Furkan Gürdoğan - 22102960

Introduction

In this project, a generative style transfer model that alters images to resemble paintings in the styles of various painters will be implemented.

Background Information

Style transfer is a method of altering the appearance of an image to resemble the style of another one. First suggested by Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge, it consists of representing the style and the content of an image separately through the utilization of convolutional neural networks (CNNs), the main idea is to minimize a loss function that includes the individual losses between contents and styles of two images. The works of Gatys, Ecker, and Matthias were improved by Johnson later, with the proposal of Fast Neural Style Transfer [1]. Today, the most widely used method in style transfer is Generative Adversarial Networks (GAN).

The GAN is a generative model used for unsupervised learning. It consists of two CNNs, namely the "generator" and the "discriminator", competing with each other for testing the resemblance of the generated image to the style set. Unlike generic supervised learning methods, in GAN the generator is not trained to minimise the distance between a specific image within the dataset and the generated image, rather it is trained to fool the discriminator trying to detect the generated image from the style dataset. There exist different approaches used to improve the performance of GANs, CoGAN, SimGAN and CycleGAN are some examples found via literature research, one or more of which may be used for the project [2].

Models

The CycleGAN model will be implemented as the main model. It was first introduced by Jun-Yan Zhu et al. in 2017 [2]. The architecture consist of two discriminator and generator pairs namely G & F as generators and D_x & D_y as discriminators. The model includes two mappings:
G: X→Y , translation of images from source domain (X) to target domain (Y)
F: Y→X , translation of images from target domain back to source domain

The discriminator D_x aims to distinguish images between $\{x\}$ and translated {F (y)}, similarly D_y distinguishes between {y} and {G(x)} [2]. The model adjusts itself based on two types of losses: the adversarial loss and the cycle-consistency loss.

Loss Functions

Adversarial Loss

It is based on binary cross-entropy (log loss) and is utilized both for discriminator and generator calibration.

- **Discriminator's Loss**

  The discriminator is trained to maximize the probability of correctly classifying real images and fake images, the adversarial loss for the discriminator can be expressed as follows.

$$L_D = -E_x \sim p_{data}(x)[logD(x)] - E_{z \sim p(z)}[log(1 - D(G(z)))] \quad (1)$$

Where:

D(x): Discriminator's output for real data x (preferably close to 1).

D(G(z)): Discriminator's output for generated data G(z) (preferably close to 0).

z: Random noise input for the generator, sampled from a prior distribution (e.g., Gaussian).

- **Generator's Loss**

  The generator is trained to fool the discriminator into classifying the generated data as real. The adversarial loss needed to be minimized for the generator is as follows:

$$LG = -E_{z \sim pz}(z)[logD(G(z))] \quad (2)$$

Where:

G(z): The generator's output (the fake data).

D(G(z)): Discriminator's prediction on the fake data

According to the loss functions above, adversarial training can learn mappings producing outputs with target domain alike distributions, but with large network capacity, it may map input images to random permutations of target images. In other words adversarial losses alone cannot ensure that a specific input is mapped to a corresponding output, as the mapping may not be unique or consistent. For the further reduction of possible mapping functions of the same image, it is argued in [2] that the mapping functions should be style consistent.

Style Consistency Loss

It minimizes the dissimilarities between the style the style of the generated image and a reference style, often defined by collection of images (e.g., a painting by Van Gogh). It aims to ensure that the transformation consistently applies the desired style throughout the image. The loss can be expressed as

$$L_{style} = \sum_{l} ||G_l(F(G(x))) - G_l(y_{style})||^2 \quad (3)$$

Where:

G_l is the Gram matrix of the feature maps at layer l,

F(G(x)) is the generated image's feature representation

y_{style} is the style reference image

According to the flow of the project additional loss function may be used such as identity loss, perceptual loss or LSGAN loss.

As an integral part of the implementation of the CycleGAN architecture hyperparameter tuning will be done with validation sets. A brief list of hyperparameters is as follows:

- Learning Rate
- Number of Epochs
- Batch Size
- Optimizer
- Cycle Consistency Loss Weight (λ)

As the project proceeds the hyperparameters stated above may be modified according to the requirements of the model.

Evaluation Metrics

As an evaluation metric, ArtFID, that builds on the Fréchet Inception Distance (FID) metric. FID is a metric that evaluates the realism of generated images by comparing their feature distributions with real images, by comparing the mean and covariance of the features.

ArtFID uses the same idea, ensuring content preservation using Learned Perceptual Image Patch Similarity (LPIPS), which is a metric that compares images by their feature representations in a CNN, and style matching using FID. LPIPS uses Euclidian distance or cosine similarity after the features are extracted [3,4,5].

Data Sets

For the style image and content sets we plan to use the WikiArt dataset of HugGAN Community [6], and the COCO dataset [7], respectively.

The WikiArt dataset includes 81,444 works of art by various artists and in various styles taken from wikiart.org. Each image has three features including "artist", "genre", and "style". The "artist" feature has 129 classes, which are labels that include the name of the artist. The "genre" feature has 11 classes such as "landscape", "portrait", "religious", etc. There are 27 classes for the "style" feature, which include "impressionism", "romanticism", and "realism", and so forth.

To reduce the computational power we will chose the following artists and implement style transfer with the art works of the following artists only:

- Vincent Van Gogh
- Salvador Dali
- Pablo Picasso

COCO stands for “Common Objects in Context”, and it is a dataset that includes images of common objects and is used in computer vision applications. Since this dataset is quite large it may require too much computational power. We may use COCO dataset just for validation and use another common object dataset for training or we may use portion of COCO dataset.

References

[1] L. A. Gatys, A. S. Ecker, and M. Bethge, “A Neural Algorithm of Artistic Style,” arXiv:1508.06576v2 [cs.CV], Sep. 2015. Available: https://arxiv.org/abs/1508.06576v2 .

[2] A. Vaswani et al., "Attention is all you need," arXiv, vol. abs/1706.03762, 2017. Available: https://arxiv.org/pdf/1703.10593

[3] T. He et al., "Learning grounded video-language representations from synthetic data," arXiv, vol. abs/2312.09008, 2023. Available: https://arxiv.org/pdf/2312.09008

[4] D. Guo et al., "Vision transformer for small-size datasets," arXiv, vol. abs/2207.12280, 2022. Available: https://arxiv.org/pdf/2207.12280

[5] “Loss Functions in Machine Learning,” Metaphysic.ai, Accessed: Oct. 21, 2024. [Online]. Available: https://blog.metaphysic.ai/loss-functions-in-machine-learning/ .

[6] Hugging Face, "Wikiart dataset viewer," Accessed: Oct. 19, 2024. [Online]. Available: https://huggingface.co/datasets/huggan/wikiart/viewer/default/train?p=1&row=101

[7] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," Accessed: Oct. 19, 2024. [Online]. Available: https://cocodataset.org/