UKRAINIAN CATHOLIC UNIVERSITY

MASTER THESIS

Color and style transfer using Generative Adversarial Networks

Author:
Andriy KUSYY

Supervisor:
Dr. Rostyslav HRYNIV

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2019

Declaration of Authorship

I, Andriy KUSYY, declare that this thesis titled, "Color and style transfer using Generative Adversarial Networks" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"Colors, like features, follow the changes of the emotions. "

Pablo Picasso

UKRAINIAN CATHOLIC UNIVERSITY

Abstract

Faculty of Applied Sciences

Master of Science

Color and style transfer using Generative Adversarial Networks

by Andriy KUSYY

In this work, we present an end-to-end solution for an image to image color and style transfer using Conditional Generative Adversarial Networks. Nowadays photo editing industry is growing rapidly, and one of the crucial issues is recoloring and restyling of individual objects or areas on images. With a fast advancement of deep segmentation models, getting a precise segmentation mask for an area on a picture is no longer a problem although unsupervised restyling and recoloring of the object with complex patterns is still a challenge. The proposed model is a state-of-the-art regarding visual appearance and provides high structural similarity.

Acknowledgements

I want to thank Orest Kupyn for enormous help with research questions related to Generative Adversarial Networks as well as Alexey Ropan for his support in Conventional Computer Vision.

Furthermore, I would like to thank WANNABY for provided computational resources and Ciklum for the scholarship, that covered most of my tuition fees.

Finally, I would like to thank Oleksiy Molchanovskyi and Rostyslav Hryniv for their supervision and guidance during the whole process.

Contents

List of Figures

List of Tables

List of Abbreviations

GAN	**G**enerative **A**dverserial **N**etwork
cGAN	**c**onditional **G**enerative **A**dverserial **N**etwork
CNN	**C**onvolutional **N**ural **N**etwork
VAE	**V**ariational **A**uto **E**ncoders

Chapter 1

Introduction

What is an image?

Generally, a digital representation of the visual appearance of physical objects could be considered as an image. From such a broad definition it follows that an image could have an extremely complex structure and content. Looking at an image, we could extract meaningful information and various structures. Illustrative examples could be objects and interactions between them, edges and areas, colors and their deepness

In the last decade, with the rapid advancement of automated image processing algorithms and models, computers become more able to extract meaningful insights from images. Notably, a lot of essential achievements were made in the area of objects detection and segmentation. Combining that with the latest outbreaks in the area of augmented reality and current computational capabilities, we are entering a stage, where we could extract, modify and replace parts of images in the real time.

With that in mind, this work addresses several research and industry oriented goals.

1.1 Research Goals

First of all, to build an efficient image-to-image color and style transfer model, which is capable of transferring complex patterns while preserving original semantic characteristics in any domains. The model should demonstrate competitive quantitative and qualitative results as well as minimum computational resources.

Secondly, we were not able to find a suitable dataset for such deep learning model training. Therefore, in this work, we are addressing possible options to overcome this limitation and generate such dataset.

Finally, we are aiming to provide an extensive comparison of possible approaches to the image-to-image transfer and their applications.

1.2 Industry Related Goals

From the more applied point of view, this work aims to create and describe an efficient pipeline for performing color and style transfer for images or their parts. We are reviewing in details such aspects of the pipeline as objects detection and segmentation, images and objects classification by style and color characteristics, and generative models for style and color transfer.

Chapter 2

Related Works

Color and style transfer has become a hot field in the recent decade. Such a rapid expansion the area is caused mostly by increasing computational capabilities, an introduction of general-purpose computing on graphical processing units as well as photo and video technology advancement. Currently, there are dozens of research challenges in various sub-areas, therefore, this chapter has two main objectives. First of all, to give a decent overview of the field and secondly, find current researches' place in it.

2.1 Colorization

Colorization is a well-studied sub-section of the color and style transfer problems set, where an input image is mapped from grayscale to a multicolor domain. As there is an infinite number of ways to colorize a single black-and-white image, creating a colorful image is just half of a process, there is always a need to be able to choose the best colorization. Therefore, for solving mentioned above challenges, were invented multiple techniques, which could be roughly divided into two main groups: user-guided and collection-guided colorization. The first group of methods relies on some hints from users about regions' colors, while the second learns priors from an extensive collection of images.

2.1.1 User-guided Colorization

Many first attempts to automate black-and-white images colorization were relying on some user provided hints about the images color distribution. First class of methods rely on user input color strokes[Levin, Lischinski, and Weiss, 2004, Qu, Wong, and Heng, 2006] Those methods are mostly working with YUV color space and had a constraint that two neighboring pixels should have similar colors if their intensities are similar Levin, Lischinski, and Weiss, 2004 Algorithms are solving an optimization problem of minimization a color difference of two neighboring pixels with similar intensities. Therefore, as they rely on low-level special difference metrics for color propagation, to achieve realistic results, those algorithms require either broad user input strokes or monotonous source image.

After the mentioned above optimization problem was formulated, most of the researches were focusing on finding better pixel similarity metric to reduce the required number of user inputs. One of the last breaks through in the field was achieved using neural networks for extracting deep features for edit propagation Endo et al., 2016

As an advancement of deep learning approach for user-guided colorization, Zhang et al., 2017 propose usage of Convolutional Neural Networks to transform a grayscale source image and a set of user input hints into a colorful version of the source image.

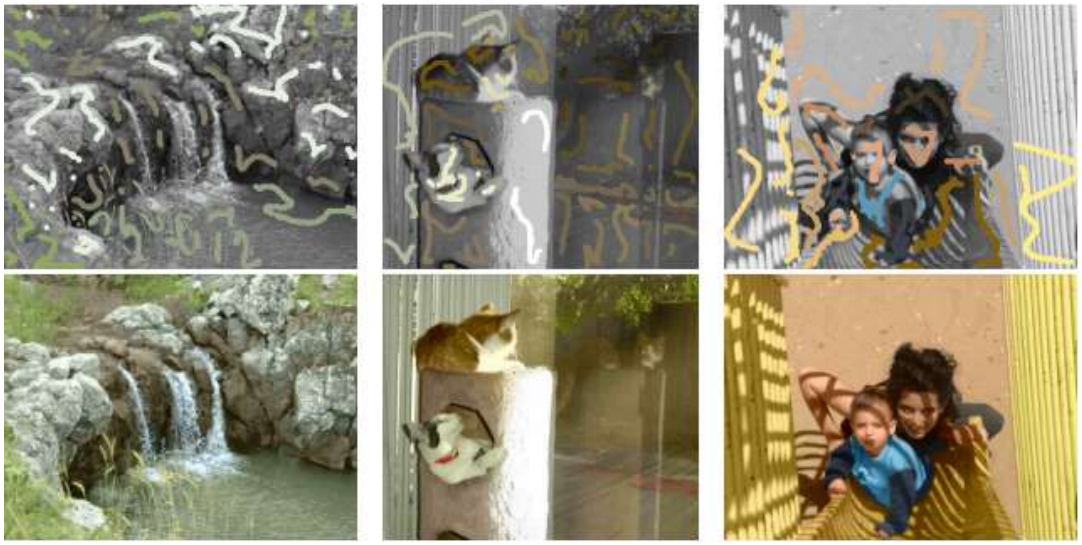

FIGURE 2.1: User-Guided Colorization example by Levin, Lischinski, and Weiss, 2004. The first row: grayscale input images with color strokes, the second one: results of the colorization

In the training process Zhang et al., 2017 generate a set of input hints along with a black-and-white version of an image from a colorful ground truth picture and then train a CNN to match that picture. Importantly, authors also introduce two ways to provide hints to the network. The first one is Local Hints Network - a set of input colors for some image's pixels and the second one - Global Hints Network - when the model learns a color distribution for a source image from another "style" image.

2.1.2 Automatic Colorization

In the automatic colorization algorithms, an input grayscale image is colorized using statistics from a single style image or a collection of style images. Very first color transfer methods provided a semi-automatic process, when an algorithm, supplied with similar domain reference image[Gupta et al., 2012, Irony, Cohen-Or, and Lischinski, 2005] or images collection [Morimoto, Taguchi, and Naemura, 2009], could extract and apply the reference color distribution to an input image. Although those algorithms are suitable for colorizing images using with really content-similar references, they are having troubles with complex structures. To perform a transfer for non-trivial correspondences, those algorithms require a mapping between a source and a reference images correspondent areas [Irony, Cohen-Or, and Lischinski, 2005].

Most of the recent breaks through in the automatic colorization field were achieved by utilizing deep learning capabilities and training on a large scale multidomain image collections [Larsson, Maire, and Shakhnarovich, 2016, Iizuka, Simo-Serra, and Ishikawa, 2016]. An essential advantage of those methods is producing multivariate outputs by generating color probability histograms for pixels. When for some objects, we could predict a color with high certainties, like orange for orange, there are objects, which could have multiple variants, like green and red apples. Depending on the style image or image collection, we would get different histograms, and therefore after merging them, we could get more diverse results. As we can see on the 2.2 due to training on an extensive collection of images, the model could with high

FIGURE 2.2: From left to right: input grayscale image, automatically generated example, and ground truth. [Larsson, Maire, and Shakhnarovich, 2016]

accuracy propagate lemons' color, although it has difficulties with the background and clothes' colors.

Predominant majority of the colorization methods, we covered so far, were working with various similarity metrics to establish conformity between some source image area and similar areas on a style image or images. Another family of models tries to learn mappings between semantic characteristics of images(objects, materials, etc.) and colors Chia et al., 2011. Afterward, those semantic characteristics are extracted from a source image and previously learned mappings are applied to them.

2.2 Color and Style Transfer

Image Colorization is only a small part of the style, and color transfer problems set, which includes but not limited to: cross-domain style transfer, photo enhancement, image blending, image denoising, and inpainting. As the scope of this work is mostly related to color transfer, we will make an accent on cross-domain style transfer works and their applications.

2.2.1 Classic Methods

For a long time color transfer for individual objects is dominated by conventional machine learning methods mostly based on Histogram Matching Algorithms like Reinhard et al., 2001 or Beigpour and Weijer, 2011a. Those methods are working effectively for recoloring purposes when we are dealing with monochrome colors. Reinhard et al., 2001 achieved a few remarkable results, which made their model a useful technique for a large set of transfer problems. First of all, for image representation, instead of an RGB, was proposed a new color space $l\alpha\beta$ As Ruderman, Cronin, and Chiao, 1998 suggested in their studies, it minimizes correlation between channels for many natural scenes, which corresponds to human visual perception. Secondly, $l\alpha\beta$ color distributions of an input image and a style image was merged. In

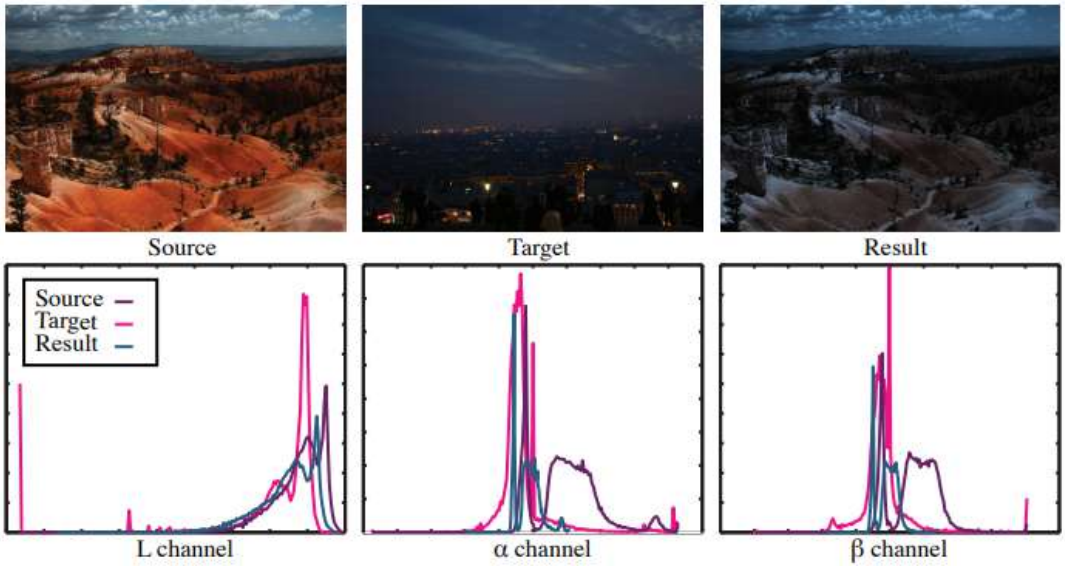

FIGURE 2.3: A pair of source and target images as well as their resulting output after using the colour transfer technique by Reinhard et al., 2001 are shown in the top row. The corresponding histograms are shown at the bottom for the three channels of the L colour space [Reinhard and Pouli, 2011]

the simplest case, as can be observed on the 2.3, only mean and a standard deviation of a source image a shifted towards the style one

The main disadvantage of the method is the dependency on the images similarity. Although semantic segmentation could partially solve the problem, by performing transfers between semantically similar areas separately, it is still failing significantly different images. Furthermore, from the mathematical formulation, the algorithm is limited in making radical changes or applying complex patterns from a style image to the source one.

Many of the issues were addressed in the number of subsequent works, in particular, Tai, Jia, and Tang, 2007 proposed adding soft color segmentation, while Pitié and Kokaram, 2007 introduced minimal displacement mapping. Those advancement allowed to achieve an active single attribute transfers for photos, like famous time of the day change problem Shih et al., 2013

One of the critical challenges, in the example-based style transfer, is selecting a single "style" image, as result dramatticaly depends on it. The problem is to choose a maximum semantically similar picture, which would be highly representative of the style. This question is widely addressed in Lee et al., 2015, where novel techniques of selecting a style image and performing a global style transfer. Many of the current works pipeline steps were inspired by Lee et al., 2015 ideas — notably, the idea of using semantic similarity metrics and clustering the input collection by style characteristics to be able to perform unsupervised learning.

Although, when styles are significantly diverse, and we need to do a notable transformation for composite objects, example-based techniques show insufficient performance and tend to acquaint example specific style artifacts. The particular challenge is illuminance and glares, which are typical for glossy surfaces. Those issues were addressed in car recoloring example in Beigpour and Weijer, 2011b, but the proposed classical solution requires precise carcass segmentation and is limited to monochrome coloring. Our suggested method for a cross-domain transfer aims

FIGURE 2.4: Example of a color transfer with manual instance segmentation by Luan et al., 2017

to solve challenges in this particular area.

2.2.2 Deep Learning

While traditional methods could solve many challenging tasks, they are unable to reflect high-level features effectively. An example of such an essential element could be an artistic style. One of the first successful results of an artistic style transfer was achieved in Gatys, Ecker, and Bethge, 2015 by constructing a loss function as a sum of two separate terms. The first one for the image content (minimization of the content difference between a source image and the generated one) while the second one for the style(minimization of the style difference between a target image and the generated one). Although even when style and content are successfully transferred, the results are not photorealistic, what makes the model unsuitable for photo style transfer. Later Luan et al., 2017 achieved more photorealistic results by designing a loss function that constrains the transformation from the source to the target image to be locally affine in a color space. Also Luan et al., 2017 use semantic segmentation model by Chen et al., 2016 to establish correspondences between images. Various semantic regions are detected on the source image, and then their analogs are determined on the target image. For instance, the source image contains sky, car, and grass; then the target one would be searched for those classes. Our work heavily utilizes the idea of instance segmentation usage 2.4 addressed in Luan et al., 2017, although we include automatic methods instead of manual segmentation.

The single-reference methods highly depend on the quality and choice of the style image, therefore, with the advancement of neural networks, were created methods capable of aggregating and transferring the style from multiple photos to the source one. Most of the methods are starting with an extensive collection of images, which are later classified or clustered into style groups by color or semantic characteristics Lee et al., 2015.

2.2.3 Generative Adversarial Networks

In the previous chapter we did an extencive overview of relevant challenges in image-to-image translation area and their partial or full solutions. Summarizing the main

Although Convolutional Neural Networks can learn complex style and color mappings, they require designing specific loss functions for separate problems. This reason makes them hard to use for cross-domain style transfer with arbitrary domains nature. Luckily, Generative Adversarial Networks are capable of learning those loss functions from data. We are reviewing GANs and cGANs in more details in **??**, as their understanding is crucial for this work. The idea of image-to-image translation with Conditional Adversarial Networks and corresponding pix2pix framework is addressed by Isola et al., 2016. Researchers achieved an outstanding result

FIGURE 2.5: Horse to zebra from unpaired dataset by Zhu et al., 2017

by creating a useful framework for many image-to-image translation problems including but not limited to day2night, edge2photo, and aerial2map. However, the proposed structure requires paired inputs, which are rarely available. This problem was partially solved in Zhu et al., 2017 by the introduction of Cycle-Consistency Loss. If we have two domains A and B, the generator function G learns to generate B-like images from A sources, but as it appears, the network tends to disregard the source image and generate any B-like ones. To overcome this challenge, Zhu et al., 2017 propose to train G, not only to create from A sources fake B images but also to reverse those fake B images back to domain A. As the result; we should get the original A image. This powerful idea helps to overcome unpaired data problem. As a result, we could generate zebra images from horse images without paired input dataset 2.5

Though, another problem of such a framework is the requirement to train a separate model for each color/style transfer case. There were two main approaches to solve this issue. The first one, addressed in Chang et al., 2018 is to supply the generator not only with source image but also, with a corresponding target image. However, this approach also requires the introduction of an additional auxiliary discriminator and due to increasing input dimensions, the memory requirement for model training increase significantly. The second one was described by Choi et al., 2017 proposes a multi-domain transfer method by changing the training strategy and training a single network for multiple domains using multiple datasets. In our work, we adopt proposed "star topology" for our model.

Chapter 3

Dataset and preprocessing

In the previous chapter, we did an extensive overview of relevant challenges in the image-to-image translation area and their partial or full solutions. Summing up, the main challenges are building multi-domain transfer models, producing photo-realistic results, processing images with visual artifacts like reflections or shadows. Furthermore, recently high resonance got various image modification techniques, especially face swap [Chang et al., 2018, Dale et al., 2011] or background change Aksoy et al., 2018. Those models include segmentation and modification of particular image area, without changes to others.

Inspired by those ideas, we propose a framework for effective image-to-image in particular object-to-object color and style transfer. Having an image of a red car in some context, we would like to be able to change its color or paint it with an intricate pattern, using a single model.

In the current and the following chapters, we are reviewing necessary for successful color/style transferring steps. As we are starting with an unclassified database of multiple objects with various colors styles and in many contexts, we cannot begin style transfer right away. This chapter aims to guide through preprocessing steps, that we are taking to transform random images set into a dataset suitable for training a transfer model.

3.1 Dataset

To our knowledge, there is no open source data set suitable for deep learning model training, which would include objects in various colors within the same context. Therefore, we decided to start from scratch. We have chosen Carvana cars dataset Carvana, 2017 as our base. It contains images of 318 cars from 16 angles and their instance segmentation masks. Furthermore, it includes pictures of 6254 additional vehicles photos from the same angles but without segmentation.

The base dataset choice was driven by various reasons. First of all, cars are ordinary objects, which appears in multiple contexts. Secondly, at the same time, cars are complex objects with the composite structure. Thirdly, vehicles surface is glossy and therefore, pictures would contain numerous blinds, reflections, and shadows. Those reasons make vehicles hard challenge for most color and style transfer model, so our motivation is to build a model, which would effectively work for them and therefore for other cases as well. Moreover, Carvana train dataset contains precise segmentation masks, which could be used to train a state of the art segmentation model and extend the dataset with additional cars and colored patterns.

FIGURE 3.1: Random Samples from Carvana Dataset

3.1.1 Detection and Segmentation

The final goal of our pipeline is to transform a particular image area, in our case, we would like to change the only appearance of a car on a picture. Therefore, a vehicle needs to be detected and sequentially segmented from the background.

There are two main approaches to segmentation. *Semantic segmentation* methods aim to assign some class label to each pixel of an image (pixel-wise classification), in contrast, *instance segmentation* requires, first of all detecting all the objects on the image and precisely segmenting them. So basically, after *semantic segmentation*, we would get a class assigned for each pixel, whether it's a car, background or any other type, however if there are multiple cars on the image, all the pixels would get the car class, while with *instance segmentation* there would be classes car1, car2, etc.

As our dataset specifics suggests, for a single image, we have only one car, so it does not matter, which segmentation approach we are using. In general, if images could contain multiple objects of interest the *instance segmentation* would be required.

One of the state-of-art solutions in instance segmentation is Mask R-CNN He et al., 2017 - an effective detection and segmentation framework build as an advancement of Faster R-CNN Ren et al., 2015

Although, Carvana car segmentation task, could be considered as a binary segmentation of a car and background, therefore in the actual Kaggle Carvana Segmentation Competition U-Net Ronneberger, Fischer, and Brox, 2015 architecture for *semantic segmentation* has shown impressive results.

Due to that reason for our pipeline, we trained the U-Net model for a binary segmentation. The vanilla U-Net has a standard encoder-decoder architecture, which could be reviewed in details on 3.2.

Originally U-Net was designed for biomedical images segmentation, but it appears to be useful for many other tasks, even for generative models so that we would address it again in the next chapter. For the car segmentation task, we used the U-Net architecture with pretrained on Imagenet VGG-11 Simonyan and Zisserman, 2014 as an encoder. For the learning rate, we used Adam optimizer, with cyclic learning rate. Many of those ideas were inspired by the winning solution of the competition Iglovikov, 2017

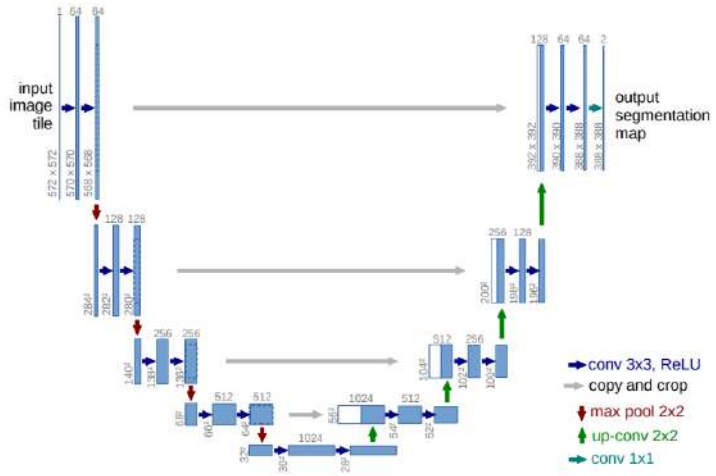

FIGURE 3.2: Vanilla U-Net architecture Ronneberger, Fischer, and Brox, 2015

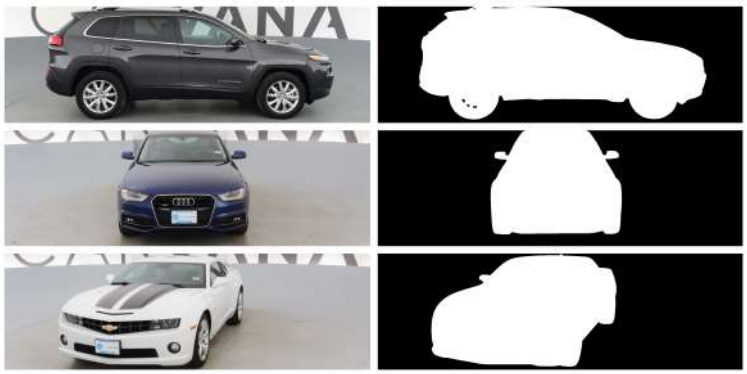

FIGURE 3.3: Carvana segmentation results

As a loss function for the network was used a combination of a Cross Entropy Loss and a Dice coefficient, as it makes predictions closer to boundaries and therefore we get more precise segmentation.

$$LOSS = BCE - ln(DICE)$$

$$DICE = \frac{2\sum_i y_i p_i}{\sum_i y_i + \sum_i p_i}$$

$$BCE = -\sum_i (y_i ln(p_i) + (1 - y_i) ln(1 - p_i))$$

Segmentation results of the original Carvana dataset images could be found on figure 3.3

As we would review in the next chapter, an effective segmentation would allow us to use segmentation loss for the generative model and significantly improve the transfer results.

3.2 Classification and Clusterization

After successful segmentation, we extracted from images our areas of interest, but it is still an unclassified collection of images.

The problem of unsupervised models for images classification was addressed in many papers, in particular, Lee et al., 2015 proposed an effective two-step approach for images clusterization using neural networks. First of all, they are doing transfer learning for pretrained on Imagenet CNN to match pictures with similar contents using high-level semantics features and afterward performs k-means clustering based on those semantic similarity scores. Secondly, a sophisticated style metric based on chrominance and luminance is used to rank images within clusters and chose the best style representatives.

Although this approach is an ideal match for many image collections and could be utilized for other tasks solved withing proposed framework, it is unsuitable for the current dataset.

As our primary goal is to perform paint and color transfer, we need to cluster images by colors and paint styles. Unfortunately, after semantics segmentation, all the photos have close scores, and the only clusters are for car types: sedan, minivan, pickup. It is possible to extract color features using another CNN and then reuse the method, but we came up with a faster classical approach.

Having segmentation masks after the previous step, we just calculated three channels RGB color distribution of car areas and performed their classification.

Chapter 4

Proposed Generative Model

In the previous chapter, we did an extensive overview of the dataset and preprocessing steps required for generative models effective applications. The current section aims to guide through the various experiments we tried and give an in-depth explanation of the final proposed generative model.

4.1 Prerequisites

4.1.1 Generative Adversarial Networks

The concept of GAN was proposed by Goodfellow et al., 2014 and it is crucial for understanding this work and conducted experiments.

Having some training data distribution P and a sample from it A. The *generator* model G is trained to create fake samples B as if they are from the original P distribution. At the same time, the second *discriminator* model is trained to distinguish fake samples B from real distribution ones.

As can be seen on 4.1 the core idea of GAN framework is training both the *generator* and the *discriminator* simultaneously, in a competitive mode. As the model converges, the generated distribution P_G is getting closer to the original distribution P. In the vanilla implementation of the GAN framework, samples are generated from the noise vector drawn from some known distribution P_z , as a rule, $\mathcal{N}(\mu, \sigma^2)$ From the theoretical point of view, this problem could be discussed regarding min-max

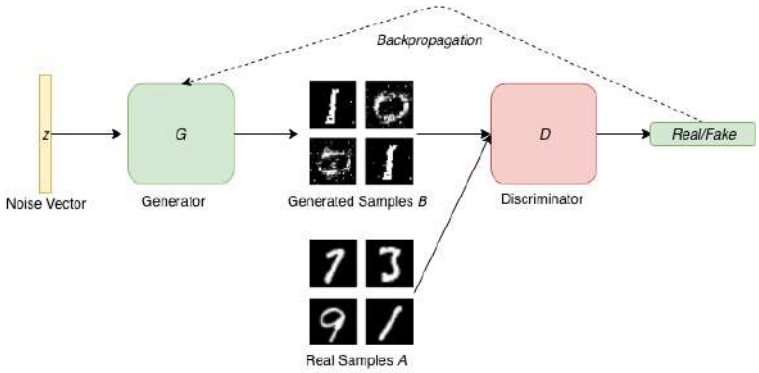

FIGURE 4.1: GAN training pipeline example on MNIST dataset. In the vanilla implementation samples are generated from $\mathcal{N}(\mu, \sigma^2)$ distribution, but later we would examine models with different input spaces

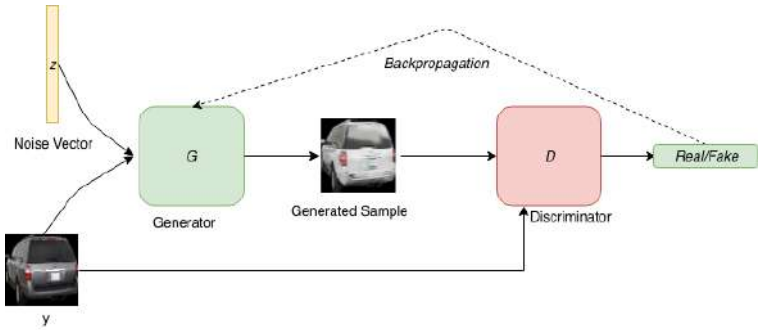

FIGURE 4.2: Pix2Pix cGAN. Both the generator and the discriminator observe the original image

optimization with a single function *V(G, D)*

$$\min_G \max_D V(G,D) = \mathbb{E}_{x\sim P(x)}[logD(x)] + \mathbb{E}_{z\sim P_z(z)}[log(1-D(G(z)))] \tag{4.1}$$

In the 4.1 the *discriminator* maximize the probability of assigning the correct label to both training examples and samples from G, while generator simultaneously tries to minimize $log(1 - D(G(z)))$ Goodfellow et al., 2014

However, there is a convergence uncertainty problem with the vanilla implementation of the GAN framework. The convergence is not guaranteed from the theoretical point of view, and even from the practical aspect, we cannot predict, whether the model would converge. Various research works addressed this issue. But mostly, they are concentrated on better loss functions Arora and Zhang, 2017 or better distribution approximation Arjovsky, Chintala, and Bottou, 2017.

4.1.2 Conditional Generative Adversarial Networks and Pix2Pix

In the vanilla implementation of the GAN, as an input for G, was used only some noise vector $z \in P(z)$ from a known distribution. Goodfellow et al., 2014 in their paper addresses another modification of the GAN framework, called Conditional Generative Adversarial Networks (cGAN). In the cGAN framework, we use not only $z \in P(z)$ as an generator input but also some extra information $y \in P(y)$. The examples of extra information could be various; it could be any constraint, target style or class variable, or even some embedding.

Isola et al., 2016 propose an implementation and extension to the cGAN framework called Pix2Pix. In the Pix2Pix framework, an image is used as a conditional variable 4.2.

Also, the loss objective is transformed to account for an original image 4.2

$$\min_G \max_D V(G,D) = \mathbb{E}_{x,y}[logD(x,y)] + \mathbb{E}_{x,z}[log(1-D(G(x,z)))] \tag{4.2}$$

Those conditions and other important advancement reviewed in details by Isola et al., 2016 allowed Pix2Pix to become one of the most useful frameworks for super-resolution, photo enhancement, image restoration, and style transfer. Also, that is the reason, why we have chosen Pix2Pix as a starting point for our generative models.

4.2 Experiments

4.2.1 Binary Collection-Based Transfer Models

Starting with an unsupervised problem of a color style transfer and inspired by the CycleGAN Zhu et al., 2017 results for horse2zebra transfer displayed on 2.5, we decided to start with a class to class transfer problem. The main advantage of CycleGAN-like approach, in this case, is lack of the requirement for paired input images.

Problem Formulation and Loss Functions Definition

Having two domains X and Y, we introduce two mapping functions $G_y : X \rightarrow Y$ and $G_x : Y \rightarrow X$ and correspondingly, two adversarial discriminator functions D_x and D_y. The generator functions $G_x(y)$ and $G_y(x_i)$ are trained to generate fake images indistinguishable from those from X and Y domains respectively. At the same time discriminators D_x and D_y are trained to distinguish $G_x(y)$ from x and $G_y(x)$ from y.

$$\mathcal{L}_{G_x}(G_x, D_x, X, Y) = \mathbb{E}_{x \sim P_{data}(x)}[log(D_x(x))] + \mathbb{E}_{y \sim P_{data}(y)}[log(1 - D_x(G_x(y)))] \quad (4.3)$$

In the CycleGAN problem setup Zhu et al., 2017 propose negative log-likelihood function 4.3 as an adversarial loss, although, as authors themselves suggest, and our experiments have shown that applying Least-Squares Loss Mao et al., 2016 leads to more stable training and better quality results. Therefore, for the final model, we are training the discriminator to maximize 4.4 and the generator to minimize 4.5. Please note, the functions are formulated only for a single generator, but as they are symmetric, formulation for the other one is trivial.

$$\mathcal{L}_{D_x}(G_x, D_x, X, Y) = \mathbb{E}_{x \sim P_{data}(x)}[(D_x(x) - 1)^2] + \mathbb{E}_{y \sim P_{data}(y)}[(D_x(G_x(y)))^2] \quad (4.4)$$

$$\mathcal{L}_{G_x}(G_x, D_x, X, Y) = \mathbb{E}_{y \sim P_{data}(y)}[(D_x(G_x(y)) - 1)^2] \quad (4.5)$$

Such problem setup provides an effective learning technique to generate images similar to those from domains, on the other hand, a generator quickly learns to ignore the input and creates merely identical to the correspondent domain images. The issue lays in the loss function definition, as it does not account for image content and therefore, uses only discriminator feedback. As the result, generated images, could have nothing in common with the originals, and simply be representations of the contrary domain features, which is unacceptable for the style and color transfer applications.

To avoid this disadvantage Cycle Consistency Loss is introduced 4.7. First generator outputs are used as inputs to the second one and vise verse 4.6. For the training steps overview, please address 4.3

$$x \in X \rightarrow G_y(x) = y' \in Y' \rightarrow G_x(y') = G_x(G_y(x)) = x' \in X' \quad (4.6)$$

Where, Y' and X' are domains of G_y and G_x generated results.

Afterwards, $L1$ norm is used as an identity loss to ensure that the cycle result x' is as close as possible of the input x

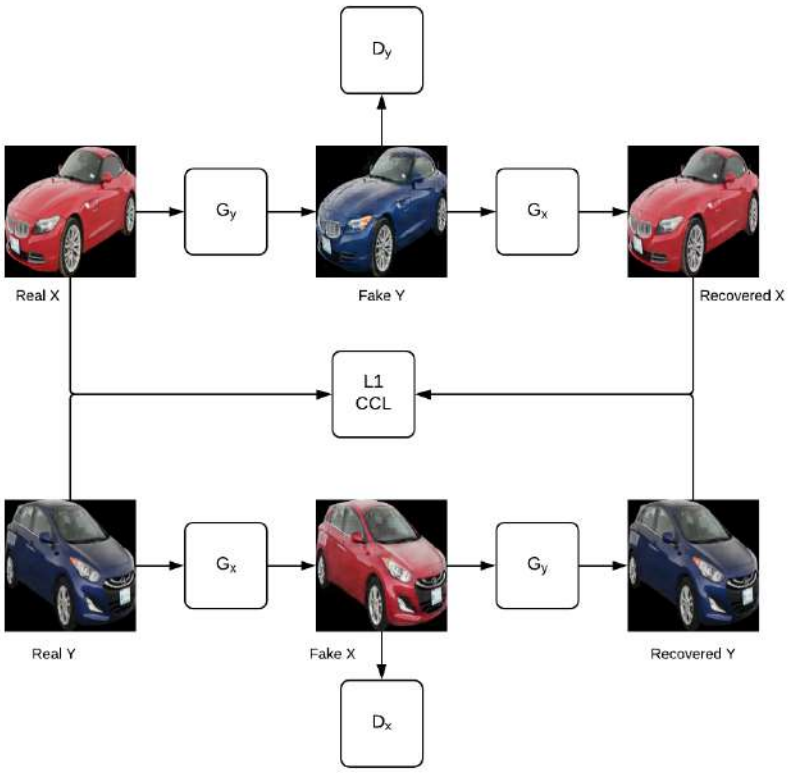

FIGURE 4.3: Binary model training step schema

$$\mathcal{L}_{ccl}(G_x, G_y) = \mathbb{E}_{x \sim P_{data}(x)}[||G_x(G_y(x)) - x||_1] + \mathbb{E}_{y \sim P_{data}(y)}[||G_y(G_x(y)) - y||_1] \quad (4.7)$$

With introduction of those three loss functions, the total loss of the model, could be defined as a combination of them 4.8

$$\mathcal{L}_{total}(G_x, G_y, D_x, D_y) = \mathcal{L}_{G_x}(G_x, D_x, X, Y) + \mathcal{L}_{G_y}(G_y, D_y, X, Y) + \lambda \mathcal{L}_{ccl}(G_x, G_y) \quad (4.8)$$

Where λ is a hyperparameter to the model, allowing to tune loss objectives importance for the case. In practice, λ value could tune the importance of the original content and the target domain style for the final result.

It is crucial to mention, although the objective is to transform only the segmented part of an image, the generator networks are taking the rectangular area as an input. Therefore, we are feeding the net with a scaled bounding area of the segmented object and reducing the the noise of background and other objects there; we calculate loss by segmentation mask.

Network Architecture and Training

The generator architecture as well as in the CycleGAN was adopted from Johnson, Alahi, and Fei-Fei, 2016 and consists of three sequential parts: encoder, transformer, and decoder 4.4.

The encoder uses stride-two convolutional layers to extract features, which later processed by nine ResNet blocks and finally restored by deconvolutions.

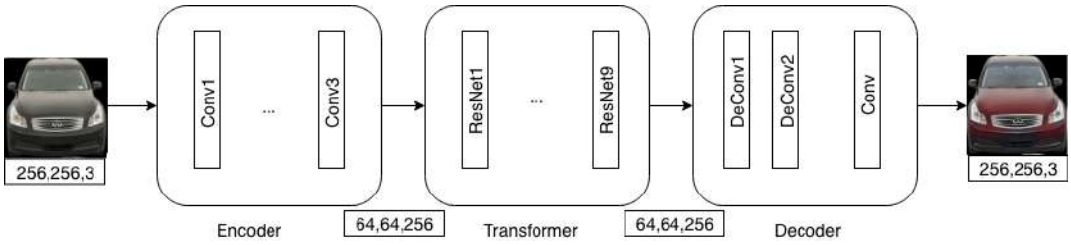

FIGURE 4.4: Generator architecture: encoding, transformation, decoding

FIGURE 4.5: Binary Collection Based-Model Results. On the left the original image, on the right - the generated one

For the discriminator, we tried both full-image and patch-level architectures. As it appears patch level architecture addressed in PatchGAN Isola et al., 2016, is much lighter and faster and still produce decent results, therefore we adopted it.

We performed training for multiple class pairs. All the images were scaled to 256×256. As a rule, we used a batch size of 4 and trained for 200 epochs. We are keeping learning rate constant for the first one hundred epochs and gradually decaying it to zero during the second hundred. As the discriminator learns much faster that generator, we were slowing it's learning by dividing an objective by a hyperparameter.

Advantages and disadvantages of binary transfer models

As could be evaluated from the results, the model quickly learned to transfer color and style, but then training slows down.

The model tends to overfit for the training data and have low generalization capabilities. Nevertheless, the main disadvantage of the binary model is a requirement to train the new model for each color pair. We made attempts for the transfer learning and fine tuning, though they speed up the process, this approach is still not scalable.

Another issue is that such problem formulations shrink the size of the training set to just two classes and therefore this approach is working only for highly represented classes but unsuitable for the less represented ones.

Based on those pros and cons of the first model, we did two more experiments by each trying to eliminate them.

4.2.2 Single-reference transfer models

The first model we reviewed here was collection based transfer, but as we mentioned the Related Works chapter, there is a whole class of single-reference based transfer models. Researching for the possible solutions, we have found that Chang et al., 2018 faced similar issues for the makeup transfer problem. Their answer was to combine both collections-based and image-based transfer techniques. We decided to adopt some of their ideas for our second model.

Problem Formulation and Loss Functions Definition

In the previous section, we used Cycle Consistency Loss to preserve the content of the original image but to apply various styles. Although the style was defined by the collection of pictures and was a generalization of their style features. In their work Chang et al., 2018 addresses an issue when there is only one or just a few images representing a style.

Therefore, X would be a space of various style images and Y is a space of no-style images. For example, in the Carvana case, Y could be defined as a class of white cars(or any other color), while $x_1, x_2, ...x_n \in X$ are cars of all the different colors. The objective is to learn mapping functions that could apply any style $x_1, x_2, ...x_n$ to the y car and at the same time, transform any of the style cars $x_1, x_2, ...x_n$ to the no-style domain Y. Please note, we are using subscripts for X domain members and not for Y to emphasize that $x_1, x_2, ...x_n \in X$ are different style pictures, while $y \in Y$ represent different images of the same style.

Similarly to the previous section, we have two generator functions G_x and G_y, although this time, they are asymmetric. $G_y(x_i)$ still takes an $x_i \in X$ as an input while $G_x(y, x_i)$ takes two images at the input layer, $y \in Y$ as a source image and $x_i \in X$ as a specific style that would be applied to y. Both discriminators are playing the same role as in the previous model.

Adversarial Loss set up is the same as for the previously, where generators and discriminators are playing min-max game 4.3, as well as the Cycle Consistency Loss is still present 4.7. The only important change is that for the G_x input, we are providing both x_i and y 4.9

$$\mathcal{L}_{G_x}(G_x, D_x, X, Y) = \mathbb{E}_{x_i \sim X}[log(D_x(x_i))] + \mathbb{E}_{y \sim Y, x_i \sim X}[log(1 - D_x(G_x(y, x_i)))] \quad (4.9)$$

However, in-depth review of such problem formulation, reveals that loss functions set up for the previous model are not enough for the effective training. As the generator is now having both the style and the source images as an input, it could quickly minimize the loss by just returning the style image all the time. To ensure

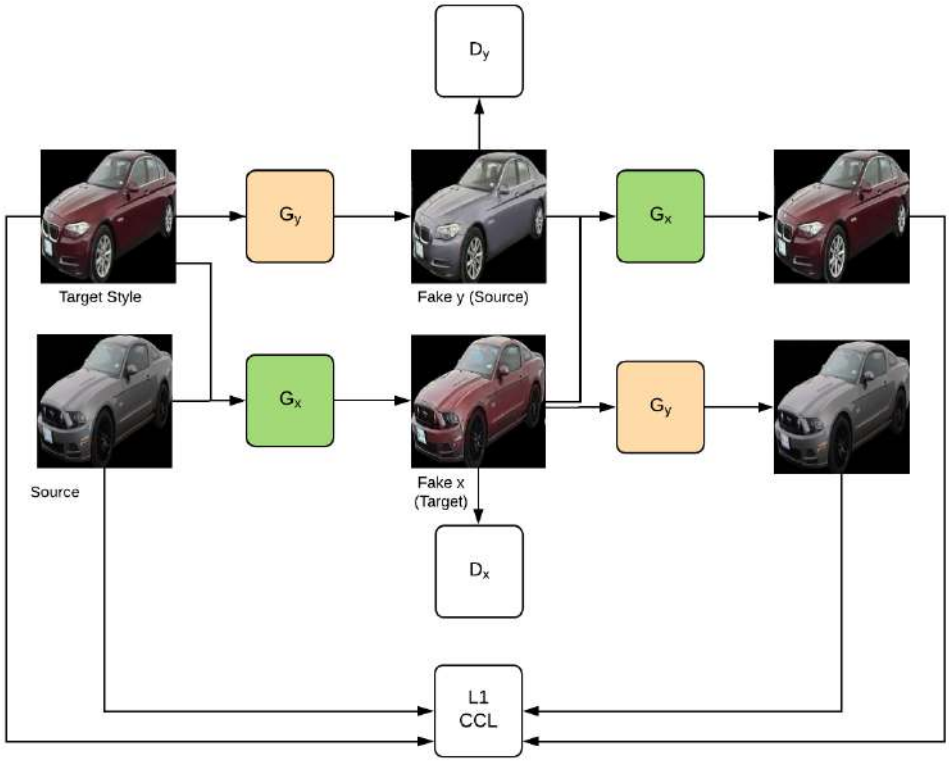

FIGURE 4.6: Image-to-image model training step. Please note that the generator and the discriminator are switched during the second step.

better style transfer results, Chang et al., 2018 propose to introduce a few more loss objectives.

First of all, the Style Loss $\mathcal{L}_{style}$, first of all, we are removing style from x_i and getting some fake no-style image y', at the same time, we are applying style x_i to some real no style image y and getting some fake styled image x'_i. Finally, we are applying fake style x'_i to the fake no-style image y' and minimizing the L1 norm of the difference with the real x_i style picture 4.10. For the detailed training procedure overview, please address 4.6

$$\mathcal{L}_{style}(G_x, G_y) = \mathbb{E}_{x_i \in X, y \in y}[||G_x(G_y(x_i), G_x(y, x_i)) - x_i||_1] \tag{4.10}$$

Nevertheless, even with the application of a Style Loss, it tends to be impossible to train an effective G_x generator to produce non-blurry results. Chang et al., 2018 in their PairedCycleGAN propose additional axillary discriminator to distinguish whether two images have the same style. Although, such a discriminator requires paired training data for an input, which was one of the main challenges for the makeup transfer problem. Luckily for the Carvana case, we had more than a single picture for each color and therefore, were able to use the classifier results to pre-train the discriminator D_s 4.7 Unfortunately, such approach would not work for style classes with the tiny number of training examples.

$$\mathcal{L}_{D_s}(G_x, D_s) = \mathbb{E}_{x_i^1, x_i^2 \in X}[logD_s(x_i^1, x_i^2)] + \mathbb{E}_{x_i \in X, y \in y}[log(1 - D_S(y, x_i))] \tag{4.11}$$

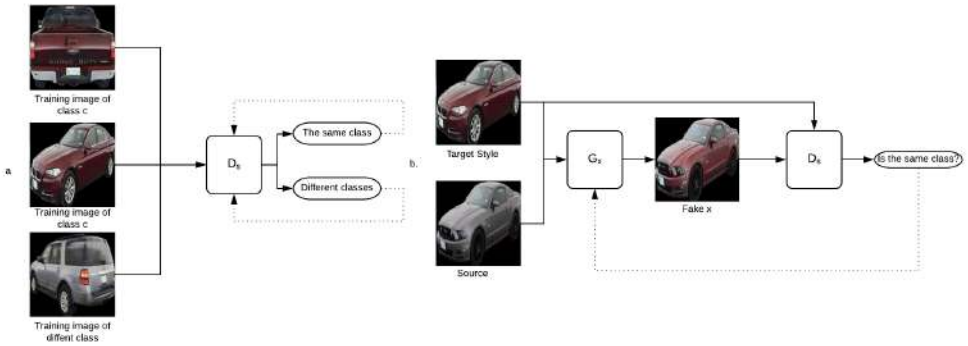

FIGURE 4.7: Image-to-image auxiliary discriminator. a. D_s raining pipeline; b. using D_s for G_x training

The total loss function 4.12, is a combination of all the objectives weight by additional hyperparameters λ_{G_x}, λ_{G_y}, λ_{D_s}, which are used to control the training pipeline and tune the degree of the transfer.

$$\mathcal{L}_{total} = \lambda_{G_x}\mathcal{L}_{G_x} + \lambda_{G_y}\mathcal{L}_{G_y} + \lambda_{D_s}\mathcal{L}_{D_s} + \mathcal{L}_{ccl} + \mathcal{L}_{style} \tag{4.12}$$

Network Architecture and Training

Regarding the generator, we are still using the same Encoder-Decoder architecture as for the first model 4.4, although now the G_x is taking two images for an input.

Chang et al., 2018 in their PairedCycleGAN propose, training generator to return not the final image in some style but only a style delta, that could be added to the no-makeup image to obtain styled one. However, such an approach is limiting the scope of the problems, where the model, could be applied and is not suitable for the Carvana case. Therefore, we are preserving the previous model logic and still generating the whole final image.

Taking into account that our dataset contains photos taken from various angles, we did two types of experiments. For the first one, we introduce the requirement that the source and the target photos should be taken from the same angle, while for the second one they could be arbitrary chosen.

For the hyperparameters λ, we have chosen addressed by Chang et al., 2018 idea of starting with the low values and increasing them step by step, as the discriminator learn and produce more trustworthy results.

Results and Challenges

As was addressed previously, by passing both the source and the target image to the input of a generator, we are adding more data for the model, but it's harder to control generator overfitting for the target image now. We partially solved this issue with additional loss objectives, but it becomes harder to control convergence of the model. Furthermore, updated generator architecture and another discriminator, has increased already significant computation requirements and limited the number of experiments, that we could run.

After some experiments, we were not able to achieve model convergence for the unaligned source-target image experiment. Regarding the alleged inputs model, the results were worse than for the binary model addressed in the previous chapter. Furthermore, the computational requirements increased significantly.

FIGURE 4.8: Single-Reference model results. From left right: source image, target style image, resulting image. One fail case is included in the second row.

The main advantage of the image to the image transfer model is a possibility to train a single model that could do one-to-many and many-to-one transfers at the same time. Although to achieve the final goal and be able to do many-to-many transfers, we still need to train multiple models.

4.2.3 Multi-domain transfer models

For the binary model, we had two classes and two generators that were learning to do the class-to-class transfer. While for the single-reference model, we had a single default style class and the domain of various style classes, so we learned how to transfer any style image to the default class and apply any style to the no-style image.

For both models, we had two generators for each one-to-one or many-to-one transfer, and the only way of scaling the model for new classes is introducing new generators or training new models, which is not a scalable approach.

There were a lot of research works aiming to eliminate those limitations. In this section, we are addressing our approach, which is based on the previous experiments and brilliant training approach discovered by Choi et al., 2017 in their StarGAN work.

Problem Formulation and Loss Functions Definition

Similarly to the single-reference transfer model, let $X = X_{c_1} \cup X_{c_2} \cup ... \cup X_{c_n}$ be a space of n various style classes $C = c_1, c_2, ...c_n$. Our goal is to train a single conditional generator function $G(x, c) : X_c \rightarrow Y_c$ that is taking an arbitrary $x \in X$ and

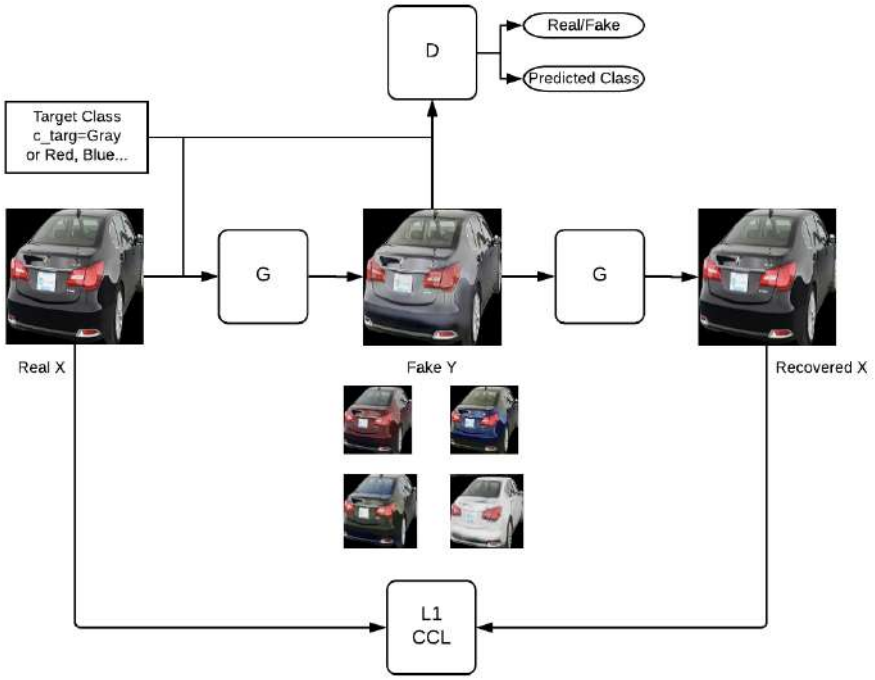

FIGURE 4.9: Multi-domain transfer architecture. The single generator G is used to transfer an image from an arbitrary source class to any target one. Cycle Consistency is preserved using L1 norm.

any class label $c \in C$ and producing fake image $y_c \in Y_c$ indistingusible by style characteristics from the images from the X_c class. Please refer to 4.13

$$\forall x \in X, \forall c \in C : G(x, c) = y_c \in Y_c \approx X_c \tag{4.13}$$

In the binary transfer problem, a discriminator was responsible for classifying fake and real images, while in the single-reference problem, we introduced additional discriminator, to classify whether two images are from the same class. For the current model, we are using two discriminator objectives. $D : x \rightarrow D_{src}(x), D_{cls}(x)$ Choi et al., 2017 distinguishing real from fake images D_{src} and at the same time classifying real images to their domains $D_{cls}(x)$. For the detailed schema of the training procedure, please refer to 4.9

On the each training step, we are feeding the generator with a source image $x \in X$ and the target domain $c \in C$.

The Adversarial Loss 4.14, is used to train the generator and the discriminator to produce realistic results and be able to distinguish real from fake images with high precision.

$$\mathcal{L}_{adv}(G, D) = \mathbb{E}_{x \in X}[log(D(x))] + \mathbb{E}_{x \in X, c \in C}[log(1 - D(G(x, c)))] \tag{4.14}$$

However, only distinguishing real from fake images, does not guarantee that generated images would be similar to the images from the domain X_c. That is why the Domain Classification Loss is introduced.

As the discriminator is having two goals, this loss consists of two separate objectives

The first one 4.15 is responsible for the fake images classification and is used to optimize the generator. We are aiming to generate images $y_c = G(x, c)$ that would be correctly classified as a c domain members.

$$\mathcal{L}_{cls}^{fake}(G, D) = \mathbb{E}_{x \in X, c \in C}[-log(D_{cls}(G(x, c)|c))] \tag{4.15}$$

Nevertheless, to classify images correctly, we need to train the discriminator before and we could use real images to do so. Therefore, we have the second objective 4.16, which is used to optimize the discriminator itself.

$$\mathcal{L}_{cls}^{real}(D) = \mathbb{E}_{c \in C, x \in X_c}[-log(D_{cls}(x|c))] \tag{4.16}$$

As we are still limited in paired input data, the only way to preserve the content of a source image is Cycle Consistency Loss 4.17. Having $x \in X'_c$ - a source image in the $c' \in C$ domain, we would like to transfer it to $c \in C$ domain and then be able to revert the transfer by applying the generator for the second time, with the original source domain as a target.

$$\mathcal{L}_{ccl}(G, D) = \mathbb{E}_{x \in X, c, c' \in C}[||G(G(x, c), c') - x||_1] \tag{4.17}$$

The final objective for the generator 4.18 and the discriminator 4.19 is a combination of mentioned above objectives wigth by hyperparameters λ_{cls} and λ_{ccl}

$$\mathcal{L}_G = \mathcal{L}_{adv} + \lambda_{cls}\mathcal{L}_{cls}^{real} + \lambda_{ccl}\mathcal{L}_{ccl} \tag{4.18}$$

$$\mathcal{L}_D = \lambda_{cls}\mathcal{L}_{cls}^{real} - \mathcal{L}_{adv} \tag{4.19}$$

Network Architecture and Training

The generator network architecture was adopted from the first model 4.4 and consists of two stride-2 convolutions, nine ResNet blocks and two transposed stride-2 convolution for upsampling. In the generator, instance normalization is was used for all layers, except the output one.

For the discriminator, we are using PatchGAN Isola et al., 2016 architecture 4.10. Using auxiliary classifier allows the discriminator to produce a probability distribution over sources and domains of images.

As well as in the previous models we are using Adam optimizer and staring with learning rate $\alpha = 0.0001$ for the first 100 epochs and decaying it to zero during the next 100 epochs. For the initial results evaluation, we were using classification error for the generated images and separately trained generator results.

Results and Challenges

The objective function dynamics, as well as the results of the generative model, could be found below. Visually, it is easy to spot the superiority of the current model in the comparison of the previous ones, regarding visual quality. Quantitative results could be found in the next chapter.

After reviewing the model and the results in details, we could highlight its main advantages.

First of all, we are building a single capable of transferring images to multiple domains and training a single model to allow using the whole dataset at once and therefore achieving better results with fewer data.

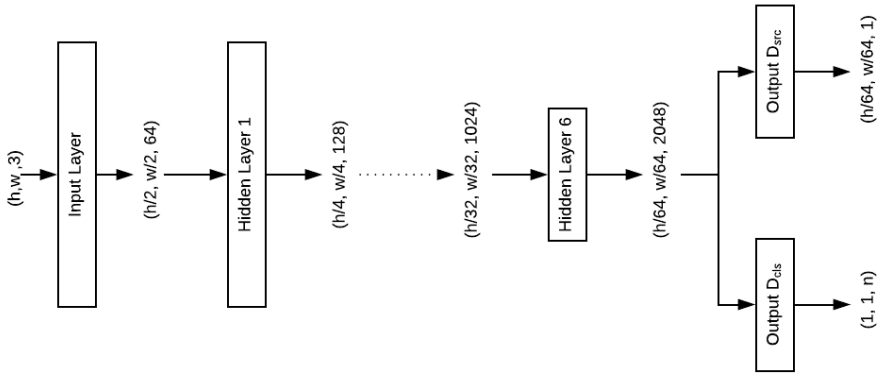

FIGURE 4.10: Multi-domain model discriminator architecture. h and w denotes images height and weight, while n stands for number of domains.

FIGURE 4.11: Multi-domain transfer model results. From left to right: original image, generated (black, blue, gray, red, white)

Secondly, the learning curve for less represented classes had improved in comparison with other models at the expense of more represented domains and generalization capabilities of a model.

The last but not least, using a single generator and discriminator, for all the classes allowed to reduce computational requirements for the model training and evaluation. Furthermore, it increased the maximum resolution of images, that could be possibly used in the model.

On the other hand, it's important to remember that it's still a collection based model, incapable of transferring style from a single image. However, results achieved by this solution could be applied to improve the second single-reference model.

Chapter 5

Results Evaluation

This chapter aims to compare qualitative and quantitative results of the reviewed in the thesis models, within the proposed pipeline, as well as compare them with baselines. As an external baseline model, we are using Deep Photo Style Transfer model by Luan et al., 2017, as it has shown impressive results for single-reference transfer and proven to be applicable for the color transfer. We are using the same pipeline for the baseline as for our models.

The task of model comparison and result evaluation for the color and style transfer models is not a trivial one as it requires image quality assessment.

For our purposes, we reviewed several approaches to solving that challenge. The initial idea was to train a separate discriminator, which would then discriminate generated images against real members of a domain. However, this approach is tricky, as for training the discriminator we were using other classes and in the end, it appears to be biased towards some of them. Another problem was that the discriminator was only able to solve the classification challenge but not the photo quality. Therefore we decided to split the problem into two parts.

As the initial benchmarking, we were using classification error for the generated images of various algorithms and baselines. For this are using the same Classifier, which we addressed in Dataset and Preprocessing chapter. For the study, we have chosen 1000 test images and classified them into six classes, manually checked the classifier result and replaced wrongly classified images. Afterward, we applied the generative models for those images and used the same classifier to the generated images. The results are indicated in 5.1

On the other hand, it is not so trivial to evaluate image quality. In many of the research works, we are referencing Choi et al., 2017, the evaluation was done by launching large scale user survey and counting Mean Opinion Score. Although this approach is one of the best, as it encapsulates real users opinion, it is not scalable for the intermediate model's evaluation due to high costs and significant time it requires.

Model	Classification Error
Binary Collection Based*	7.3%
Single Reference	11.2%
Deep Photo Style Transfer	8.9%
Multi-domain Model	5.1%
Real Images	*0.7%*

TABLE 5.1: Results Evaluation using Classifier. *Please note that result for binary collection based model is a composite of results for multiple paired models

FIGURE 5.1: Results comparison between baseline and proposed single-reference transfer model. Original(L), Proposed Method(C), Baseline(R) Luan et al., 2017

Model	NIMA Score
Binary Collection Based*	2.65
Single Reference	1.54
Deep Photo Style Transfer	3.76
Multi-domain Model	4.12

TABLE 5.2: Results Evaluation using NIMA. *Please note that result for binary collection based model is a composite of results for multiple paired models

Therefore, we have chosen an automated pretrained NIMA model for the image assessment Esfandarani and Milanfar, 2017. The model was pretrained on human scored dataset and taking an image as an input it produces a score in $[0..10]$ range. We have chosen a random subsample of 50 generated images for each model and run the NIMA scoring. As could be seen at 5.2 the Multidomain transfer model is a clear winner regarding classification and quality assessment errors.

Chapter 6

Conclusions

In this work, did an extensive overview of existing color and style transfer models and challenges, as well as present an effective unsupervised generative model which outperforms other reviewed models, as well as existing competitors in qualitative and quantitative results.

Furthermore, we provide an applied pipeline for the object-level color transfer, which solves classification, segmentation, and transfer problems. The proposed pipeline could be used to create a paired dataset of images, where the same objects have different color and style characteristics. Further, such dataset could be used for pretraining supervised model and enhancing current results.

Bibliography

Aksoy, Yağız et al. (2018). "Semantic Soft Segmentation". In: *ACM Trans. Graph. (Proc. SIGGRAPH)* 37.4, 72:1–72:13.

Arjovsky, Martin, Soumith Chintala, and Léon Bottou (2017). "Wasserstein Generative Adversarial Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, pp. 214–223. URL: http://proceedings.mlr.press/v70/arjovsky17a.html.

Arora, Sanjeev and Yi Zhang (2017). "Do GANs actually learn the distribution? An empirical study". In: *CoRR* abs/1706.08224. arXiv: 1706.08224. URL: http://arxiv.org/abs/1706.08224.

Beigpour, Shida and Joost van de Weijer (2011a). *Object Recoloring based on Intrinsic Image Estimation*. URL: http://www.cat.uab.cat/Public/Publications/2011/BeV2011.

— (2011b). "Object recoloring based on intrinsic image estimation". In: *2011 International Conference on Computer Vision*, pp. 327–334.

Carvana (2017). *Carvana Image Masking Challenge*. URL: https://www.kaggle.com/c/carvana-image-masking-challenge/data.

Chang, Huiwen et al. (2018). "PairedCycleGAN: Asymmetric Style Transfer for Applying and Removing Makeup". In: *CVPR 2018*.

Chen, Liang-Chieh et al. (2016). "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs". In: *CoRR* abs/1606.00915. arXiv: 1606.00915. URL: http://arxiv.org/abs/1606.00915.

Chia, Alex Yong-Sang et al. (2011). "Semantic Colorization with Internet Images". In: *ACM Trans. Graph.* 30.6, 156:1–156:8. ISSN: 0730-0301. DOI: 10.1145/2070781.2024190. URL: http://doi.acm.org/10.1145/2070781.2024190.

Choi, Yunjey et al. (2017). "StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation". In: *CoRR* abs/1711.09020. arXiv: 1711.09020. URL: http://arxiv.org/abs/1711.09020.

Dale, Kevin et al. (2011). "Video Face Replacement". In: *ACM Trans. Graph.* 30.6, 130:1–130:10. ISSN: 0730-0301. DOI: 10.1145/2070781.2024164. URL: http://doi.acm.org/10.1145/2070781.2024164.

Endo, Yuki et al. (2016). "DeepProp: Extracting Deep Features from a Single Image for Edit Propagation". In: *Computer Graphics Forum*. ISSN: 1467-8659. DOI: 10.1111/cgf.12822.

Esfandarani, Hossein Talebi and Peyman Milanfar (2017). "NIMA: Neural Image Assessment". In: *CoRR* abs/1709.05424. arXiv: 1709.05424. URL: http://arxiv.org/abs/1709.05424.

Gatys, Leon A., Alexander S. Ecker, and Matthias Bethge (2015). "A Neural Algorithm of Artistic Style". In: *CoRR* abs/1508.06576. arXiv: 1508.06576. URL: http://arxiv.org/abs/1508.06576.

Goodfellow, Ian J. et al. (2014). "Generative Adversarial Nets". In: *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*.

NIPS'14. Montreal, Canada: MIT Press, pp. 2672–2680. URL: http://dl.acm.org/citation.cfm?id=2969033.2969125.

Gupta, Raj Kumar et al. (2012). "Image Colorization Using Similar Images". In: *Proceedings of the 20th ACM International Conference on Multimedia*. MM '12. Nara, Japan: ACM, pp. 369–378. ISBN: 978-1-4503-1089-5. DOI: 10.1145/2393347.2393402. URL: http://doi.acm.org/10.1145/2393347.2393402.

He, Kaiming et al. (2017). "Mask R-CNN". In: *CoRR* abs/1703.06870. arXiv: 1703.06870. URL: http://arxiv.org/abs/1703.06870.

Iglovikov, Vladimir (2017). *Kaggle Carvana Image Masking Challenge*. URL: https://github.com/asanakoy/kaggle_carvana_segmentation.

Iizuka, Satoshi, Edgar Simo-Serra, and Hiroshi Ishikawa (2016). "Let there be Color!: Joint End-to-end Learning of Global and Local Image Priors for Automatic Image Colorization with Simultaneous Classification". In: *ACM Transactions on Graphics (Proc. of SIGGRAPH 2016)* 35.4, 110:1–110:11.

Irony, Revital, Daniel Cohen-Or, and Dani Lischinski (2005). "Colorization by Example". In: *Proceedings of the Sixteenth Eurographics Conference on Rendering Techniques*. EGSR '05. Konstanz, Germany: Eurographics Association, pp. 201–210. ISBN: 3-905673-23-1. DOI: 10.2312/EGWR/EGSR05/201-210. URL: http://dx.doi.org/10.2312/EGWR/EGSR05/201-210.

Isola, Phillip et al. (2016). "Image-to-Image Translation with Conditional Adversarial Networks". In: *CoRR* abs/1611.07004. arXiv: 1611.07004. URL: http://arxiv.org/abs/1611.07004.

Johnson, Justin, Alexandre Alahi, and Li Fei-Fei (2016). "Perceptual losses for real-time style transfer and super-resolution". In: *European Conference on Computer Vision*.

Larsson, Gustav, Michael Maire, and Gregory Shakhnarovich (2016). "Learning Representations for Automatic Colorization". In: *CoRR* abs/1603.06668. arXiv: 1603.06668. URL: http://arxiv.org/abs/1603.06668.

Lee, Joon-Young et al. (2015). "Automatic Content-Aware Color and Tone Stylization". In: *CoRR* abs/1511.03748. arXiv: 1511.03748. URL: http://arxiv.org/abs/1511.03748.

Levin, Anat, Dani Lischinski, and Yair Weiss (2004). "Colorization Using Optimization". In: *ACM Trans. Graph.* 23.3, pp. 689–694. ISSN: 0730-0301. DOI: 10.1145/1015706.1015780. URL: http://doi.acm.org/10.1145/1015706.1015780.

Luan, Fujun et al. (2017). "Deep Photo Style Transfer". In: *CoRR* abs/1703.07511. arXiv: 1703.07511. URL: http://arxiv.org/abs/1703.07511.

Mao, Xudong et al. (2016). "Multi-class Generative Adversarial Networks with the L2 Loss Function". In: *CoRR* abs/1611.04076. arXiv: 1611.04076. URL: http://arxiv.org/abs/1611.04076.

Morimoto, Yuji, Yuichi Taguchi, and Takeshi Naemura (2009). "Automatic Colorization of Grayscale Images Using Multiple Images on the Web". In: *SIGGRAPH 2009: Talks*. SIGGRAPH '09. New Orleans, Louisiana: ACM, 59:1–59:1. ISBN: 978-1-60558-834-6. DOI: 10.1145/1597990.1598049. URL: http://doi.acm.org/10.1145/1597990.1598049.

Pitié, F and A Kokaram (2007). "The linear Monge-Kantorovitch colour mapping for example-based colour transfer". In: pp. 1 –9. ISBN: 978-0-86341-843-3. DOI: 10.1049/cp:20070055.

Qu, Yingge, Tien-Tsin Wong, and Pheng-Ann Heng (2006). "Manga Colorization". In: *ACM Trans. Graph.* 25.3, pp. 1214–1220. ISSN: 0730-0301. DOI: 10.1145/1141911.1142017. URL: http://doi.acm.org/10.1145/1141911.1142017.

Reinhard, Erik and Tania Pouli (2011). *Colour Spaces for Colour Transfer*.

Reinhard, Erik et al. (2001). "Color Transfer Between Images". In: *IEEE Comput. Graph. Appl.* 21.5, pp. 34–41. ISSN: 0272-1716. DOI: 10.1109/38.946629. URL: http://dx.doi.org/10.1109/38.946629.

Ren, Shaoqing et al. (2015). "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *CoRR* abs/1506.01497. arXiv: 1506.01497. URL: http://arxiv.org/abs/1506.01497.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *CoRR* abs/1505.04597. arXiv: 1505.04597. URL: http://arxiv.org/abs/1505.04597.

Ruderman, Daniel L., Thomas W. Cronin, and Chuan-Chin Chiao (1998). "Statistics of Cone Responses to Natural Images: Implications for Visual Coding". In: *Journal of the Optical Society of America A* 15, pp. 2036–2045.

Shih, Yichang et al. (2013). "Data-driven Hallucination of Different Times of Day from a Single Outdoor Photo". In: *ACM Trans. Graph.* 32.6, 200:1–200:11. ISSN: 0730-0301. DOI: 10.1145/2508363.2508419. URL: http://doi.acm.org/10.1145/2508363.2508419.

Simonyan, Karen and Andrew Zisserman (2014). "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *CoRR* abs/1409.1556. arXiv: 1409.1556. URL: http://arxiv.org/abs/1409.1556.

Tai, Yu-Wing, Jiaya Jia, and Chi-Keung Tang (2007). "Soft Color Segmentation and Its Applications". In: *IEEE transactions on pattern analysis and machine intelligence* 29, pp. 1520–37. DOI: 10.1109/TPAMI.2007.1168.

Zhang, Richard et al. (2017). "Real-Time User-Guided Image Colorization with Learned Deep Priors". In: *ACM Transactions on Graphics (TOG)* 9.4.

Zhu, Jun-Yan et al. (2017). "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networkss". In: