A Seminar Report on:

# “Image Cartoonification With Neural Network”


**Prepared by : Bhagya Vinod Rana**

**Roll. No. : U19CS012**

**Class : B.Tech –III (Computer Science & Engineering) 5th Semester**

**Year : 2021-22**

**Guided by : Gautam Kumar**




**Department of Computer Engineering**

**Sardar Vallabhbhai National Institute of Technology,**

**Surat -395007 (Gujarat), India**

**Sardar Vallabhbhai National Institute of Technology,**

**Surat -395007 (Gujarat), India**

# CERTIFICATE

This is to certify that the Seminar Report entitled

**Image Cartoonification With Neural Network**

is prepared and presented by **Mr. Bhagya Vinod Rana** bearing Roll No. : **U19CS012**, 3rd Year of **B.Tech (Computer Science and Engineering)** and his work is satisfactory.

| GUIDE | JURY | HOD |
|---|---|---|
| Gautam Kumar | | COED |

# Contents

# List of Figures

# List of Abbreviations

| | |
|---|---|
| CNN | Convolutional Neural Network |
| DCGAN | Deep Convolutional Generative Adversarial Network |
| FID | Frechet Inception Distance |
| GAN | Generative Adversarial Network |
| LReLU | Leaky Rectified Linear Activation |
| LSTM | Long Short Term Memory |
| WBC | White Box Cartoonization |

**Abstract**

We propose extending the existing GAN framework with a new framework for estimating generative models via an adversarial process, as well as developing a white-box controlled image cartoonization that can create high-quality cartoonized pictures from real-world photos.

The images are broken down into three cartoon forms. The surface representation of cartoon pictures comprises a smooth surface, the structure representation relates to sparse color-blocks and flattens global information in the celluloid style process, and the texture representation represents high-frequency texture, curves, and features in cartoon images.

Our method's learning objectives are based on each extracted representation individually, allowing us to regulate and alter our framework. Through a qualitative and quantitative examination of the created samples, this study exhibits the framework's potential. Learning to Cartoonize Using White-Box Cartoon Representations is the model provided by Xinrui Wang and Jinze Yu for this project.

# Chapter 1

# Introduction

Cartoons are a popular art style that has been used in a variety of settings, ranging from print media to children's narrative. Some of the cartoon artwork was inspired by real-life scenes. Manually recreating real-life scenarios, on the other hand, may be time-consuming and needs advanced abilities.

The advancement of Machine Learning has broadened the possibilities for making visual artworks. Some well-known items have been produced by converting real-world photos into suitable cartoon scene components, a technique known as picture cartoonization.

Using the GAN framework, white box cartoonization transforms high-quality real-life photographs into outstanding cartoon visuals.

## 1.1 Aim and Objective of the Project

The primary goal of this project is to construct current cartoon animation workflows that allow artists to produce material from a number of sources. A cartoon is a popular art form that has been extensively used in a variety of settings. Some well-known items have been produced by converting real-world photos into suitable cartoon scene components, a technique known as picture cartoonization.

The need to have a thorough understanding of the Generative Adversarial Networks (GANs) Framework was one of the driving forces for the creation of this project. Interest in creating a user-friendly picture cartoonization paradigm. To broaden our understanding of recently created technologies such as CartoonGAN - Tensorflow, and W-B Cartoonization, we will attempt to build it as a web app utilising HTML/CSS and Tensorflow.js.

## 1.2 Scope of the Project

In picture improvement and augmentation, Generative Adversarial Networks (GAN) have achieved impressive results. Our current project is centred on the cartoonization of high-quality photos. This might be useful in the entertainment sector. GAN framework may also be applied in other domains including as video games, anime, and animated films.

## 1.3 Organization of the Report

- The first chapter presents a quick outline of the project's goals. It also establishes the project's scope.
- The literature study of the current system is included in Chapter 2 of the report, as well as a brief explanation of the existing system and comparisons between the CartoonGAN system and White Box Cartoonization utilising GAN.
- The issue statement is defined in Chapter 3 and a new system is proposed as a solution.
- The UML diagrams in Chapter 4 provide an abstract perspective of the system. This chapter provides a full overview of the technology and processes employed in the project's development. In addition, the hardware and software requirements, as well as the programme components, are specified.
- The approach employed in our proposed system, as well as the various designs of the components, are elaborated in Chapter 5. It also describes the mathematical ideas and methods that will be employed in the project's execution.
- The project's performance is evaluated in Chapter 6. To achieve the same, we've built up an experimental setup. Quantitative and qualitative comparisons are also provided.
- The final chapter is Chapter 7. This chapter provides an overview of the entire project.

# Chapter 2

# Literature Survey

## 2.1 A Brief History of an Existing System Comparable to Our Selected System

Cartooning is a traditional art form in and of itself, but advancements in the field of Artificial Intelligence have opened up a slew of new possibilities. Many models for creating cartoon graphics from photographs have been devised, however they all have flaws.
CartoonGAN is one of the technologies for creating cartoonized pictures, however it adds noise to the image and lowers its quality. White Box Cartoonization, on the other hand, solves these issues and produces visuals that are more accurate and crisp.

The following are the difficulties in CartoonGAN:

1. The issue of generator and discriminator stability.
2. The problem of determining the object's location.
3. The difficulty in grasping the perspective of visuals, whether 2D or 3D.
4. The difficulty in comprehending global items such as trees, flowers, and so on.

This study will be based on the Learning to Cartoonize Using White-Box Cartoon Representations paradigm established by Xinrui Wang and Jinze Yu. The major goal of this project is to construct current cartoon animation workflows that allow artists to produce material from a range of sources.

## 2.2 WBC Using GAN Framework vs. Previous System

White Box Cartoonization is based on Black Box Cartoonization, although it addresses some of the latter's flaws.

By analysing the produced dataset, certain cartoon methods, for example, pay greater attention to global palette themes and consider line sharpness as a secondary problem. Others place a high emphasis on the sharpness of items and people. Black box cartoonization algorithms struggle to deal with such a wide range of workflow requirements, and fitting the training data directly with a black-box model might have a detrimental impact on generality and stylization quality, resulting in low-quality outputs.

Researchers consulted artists and observed cartoon painting behaviour to identify three distinct cartoon image representations: a surface representation that refers to sparse color-blocks and flattens global content in the workflow, and a texture representation that reflects high-frequency texture, contours, and details in cartoon images. Each representation is retrieved via image processing modules, and a generative adversarial network (GAN) architecture is used to learn the extracted representations and cartoonize the input pictures.The weight of each representation in the loss function can be adjusted to regulate output styles.

The proposed approach outperformed three existing cartoonization methods in both cartoon quality (similarity between the input images and cartoon images) and overall quality (identifying undesirable colour shifts, texture distortions, high-frequency noise, or other artefacts in the images) in a user study with 10 respondents and 30 images.

# Chapter 3

# Problem Statement

These days, AI-powered cartoonization has a wide range of practical uses, ranging from individualised anime-style avatars to video and even fine art. Many black-box cartoonization frameworks, on the other hand, provide consumers little flexibility or adaptability when it comes to converting real-world images into cartoon sceneries.

Researchers have now presented a framework that can generate high-quality cartoonized pictures with significantly increased controllability to fulfil the needs of artists across a broader spectrum of styles and application cases. We hope to create a web application that users may feed photographs into to generate high-quality cartoonized outputs.

# Chapter 4

# Design and Implementation

## 4.1 Use Case Diagram

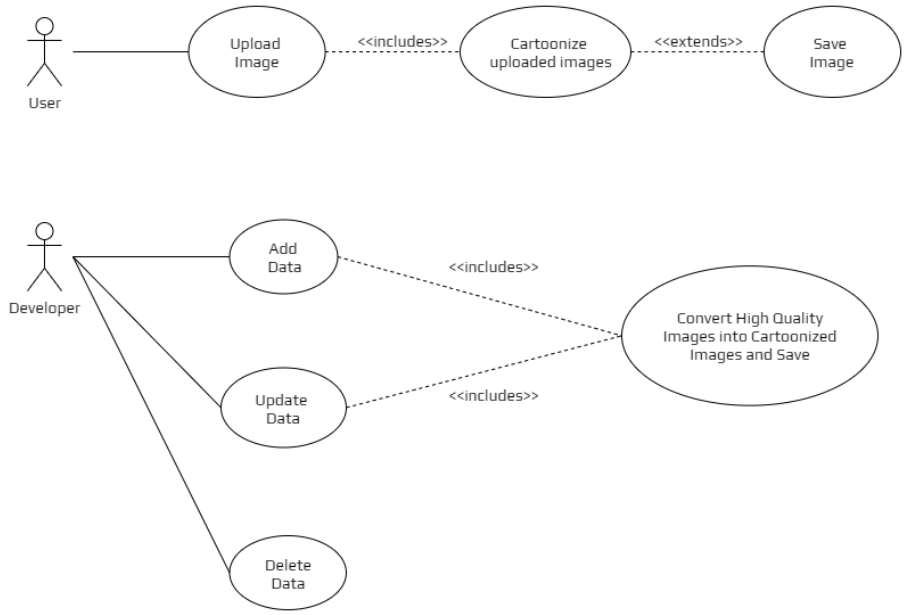


**Figure 4.1: Use Case Diagram**

Figure 4.1 Depicts the Case Study Diagram of the Project.

## 4.2 Flowchart of White-Box-Cartoonization Model

Figure 4.2 Depicts White-Box-Cartoonization Model Flowchart.

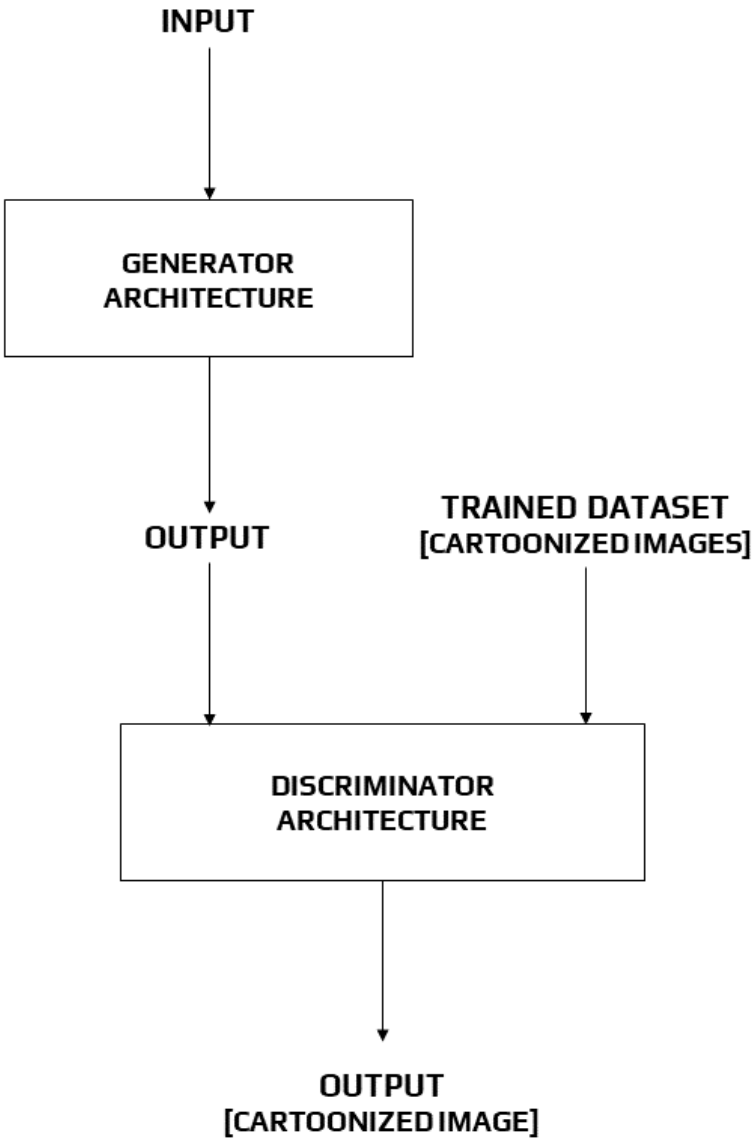


**Figure 4.2: Flowchart**

## 4.3 Architecture of WBC Model

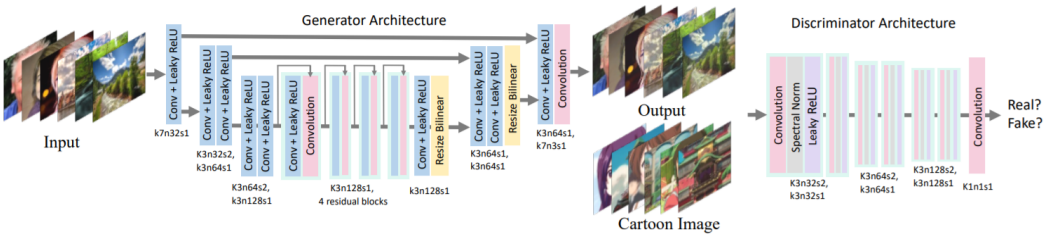


**Figure 4.3: Architecture of Generator and Discriminator**

In the diagram 4.3 above, we demonstrate the architecture of the generator and discriminator networks. The generator network is a U-Net-like network that is completely convolutional. To eliminate checkerboard effects, we utilise convolution layers with stride2 for downsampling and bilinear interpolation layers for upsampling. There are just three types of layers in the network: convolution, Leaky ReLU (LReLU), and bilinear resize layers.
This makes it simple to integrate into edge devices like mobile phones. PatchGAN is used in the discriminator network, which has a convolution layer as the final layer.

Each pixel in the output feature map corresponds to a patch in the input picture with a patch size equal to the perceptual field and is used to determine whether the patch belongs to cartoon or produced images. PatchGAN improves detail discrimination and speeds training. To enforce the Lipschitz constraint on the network and stabilise training, spectral normalisation is applied after each convolution layer (save the last one).

# Chapter 5

# Methodology

## 5.1 Introduction to Generative Adversarial Networks(GANs)

GANs (generative adversarial networks) are a fascinating new machine learning technique. GANs are generative models, which means they generate new data instances based on your training data. GANs, for example, can develop visuals that resemble photos of human faces, despite the fact that the faces do not belong to anybody.

GANs are a generative modelling approach that employs deep learning approaches such as convolutional neural networks. Generative modelling is an unsupervised learning job in machine learning that entails automatically detecting and learning the regularities or patterns in incoming data such that the model can be used to produce or output new instances that may have been chosen from the original dataset.

GANs are a clever way of training a generative model by framing the problem as a supervised learning problem with two sub-models: the generator model, which we train to generate new examples, and the discriminator model, which tries to classify examples as real (from the domain) or fake (from outside the domain) (generated). The two models are trained in an adversarial zero-sum game until the discriminator model is tricked roughly half of the time, indicating that the generator model is producing believable instances.

GANs are an exciting and rapidly changing field that fulfils the promise of generative models by generating realistic examples across a wide range of problem domains, most notably in image-to-image translation tasks such as translating photos of summer to winter or day to night, and in generating photorealistic photos of objects, scenes, and people that even humans cannot tell are fake.

## 5.2 Deep Convolutional Generative Adversarial Networks

DCGAN is an extension of the GAN architecture that uses deep convolutional neural networks for both the generator and discriminator models, as well as model and training settings that result in robust generator training.

The DCGAN is significant because it identified the model restrictions that must be met in order to construct high-quality generator models in practise. As a result of this design, a huge number of GAN extensions and applications have been developed quickly.

## 5.3 Architecture of GAN

A GAN's architecture consists of two main components: the generating network and the discriminator network. Each network can be any type of neural network, including an Artificial Neural Network (ANN), a Convolutional Neural Network (CNN), a Recurrent Neural Network (RNN), or a Long Short Term Memory (LSTM) (LSTM). The discriminator must have completely linked layers, culminating in a classifier.

The generator learns to create realistic data in a generative adversarial network (GAN). The discriminator learns to identify the generator's bogus data from genuine data by using the created instances as negative training examples. The generator is penalised by the discriminator if it produces improbable results.

When training begins, the generator produces fake data, and the discriminator quickly learns to tell that it's fake:


**Figure 5.1: Discriminator is Learning**

As training progresses, the generator gets closer to producing output that can fool the discriminator:

Finally, if generator training goes well, the discriminator gets worse at telling the difference between real and fake. It starts to classify fake data as real, and its accuracy decreases.


**Figure 5.2: Output to Fool Discriminator**


**Figure 5.3: Results**

Whole System Explained in Below Figure 5.4

The discriminator and the generator are both neural networks. The discriminator input is directly connected to the generator output. The discriminator's classification delivers a signal to the generator, which it uses to update its weights through backpropagation.

### 5.3.1 The Discriminator

In a GAN, the discriminator is just a classifier. It tries to tell the difference between genuine data and data generated by the generator. Any network architecture relevant to the sort of data it's categorising might be used.

#### Discriminator Training Data

The discriminator's training data is derived from two sources: genuine data instances, such as real photographs of individuals, and synthetic data instances. During training, the discriminator considers these situations as positive examples. The generator generates fictitious data objects. During training, the discriminator utilises these situations as negative examples.

The two "Sample" boxes in Figure 5.5 reflect the two data sources that input into the discriminator. The generator does not train during discriminator training. It keeps its weights constant while creating samples for the discriminator to learn from.

#### Training the Discriminator

The discriminator is linked to two different loss functions. During discriminator training, the discriminator disregards the generator loss in favour of focusing solely on the discriminator loss.

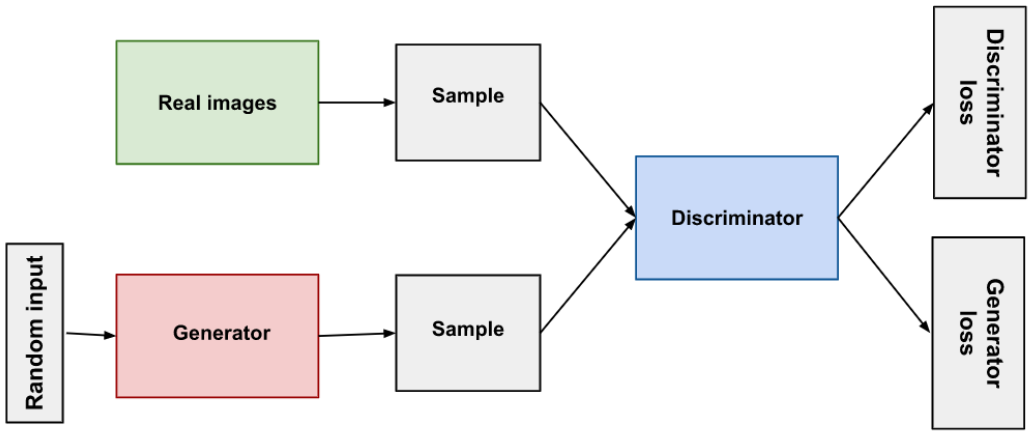


**Figure 5.4: Architecture of GAN**

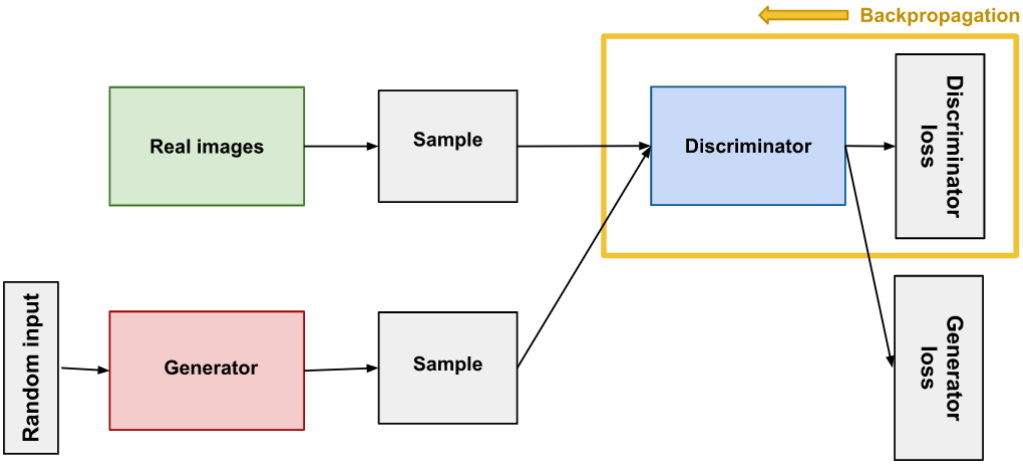


**Figure 5.5: Backpropagation in discriminator training**

During generator training, we employ the generator loss, as detailed in the following section. During discriminator training,

1. The discriminator distinguishes between actual and fraudulent data generated by the generator.

2. The discriminator loss penalises the discriminator for incorrectly categorising a genuine instance as a fake instance or a fake instance as a real instance.

3. The discriminator's weights are updated by backpropagation from the discriminator loss via the discriminator network.

### 5.3.2 Generator

By integrating input from the discriminator, the generator element of a GAN learns to produce bogus data. It learns how to convince the discriminator that its output is real.

Generator training necessitates a greater degree of integration between the generator and the discriminator than discriminator training. The random input generator network, which turns random input into a data instance, is part of the GAN that trains the generator.

discriminator network, which classifies the generated data discriminator output, which penalises the generator for failing to deceive the discriminator generator loss, which penalises the generator for failing to fool the discriminator.

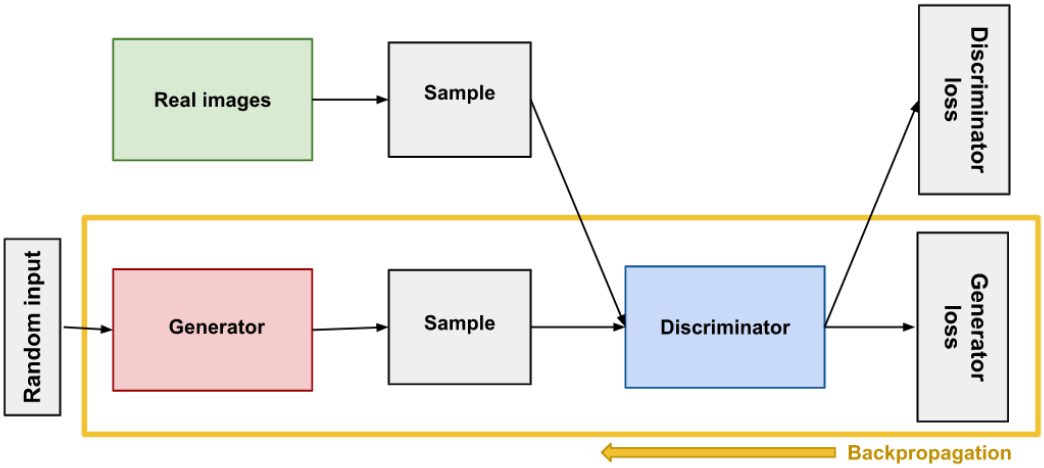


**Figure 5.6: Backpropagation in generator training**

#### Using the Discriminator to Train the Generator

To train a neural network, we change the weights of the net to decrease the inaccuracy or loss of its output. However, in our GAN, the generator is not directly related to the loss that we are attempting to change. The generator net feeds into the discriminator net, which provides the output we're seeking to influence. The generator loss penalises the generator for providing a sample that the discriminator network deems to be fraudulent.

Backpropagation corrects each weight by estimating the weight's influence on the output, or how the output would change if the weight were modified. The impact of a generator weight, on the other hand, is determined by the impact of the discriminator weights into which it feeds. As a result, backpropagation begins at the output and flows back into the generator via the discriminator.

At the same time, we don't want the discriminator to change while the generator is being trained. Attempting to strike a moving target would make an already difficult situation considerably more difficult for the generator. As a result, we train the generator using the following procedure:

1. Take a random noise sample.
2. Generate generator output using sampled random noise.
3. Determine whether the discriminator's output is "Real" or "Fake".
4. Determine the loss resulting from discriminator categorization.
5. To acquire gradients, backpropagate via both the discriminator and the generator.
6. Change just the generator weights using gradients.

## 5.4 Pros of GAN Framework

Exceptional outcome in terms of content created and realistic visual content; may also be modelled for other applications such as picture restoration, etc.

## 5.5 Cons of GAN Framework

Despite GAN's success, the potential of such a framework has several limitations. Training is difficult since it requires a large amount of computation. "Realistic," yet not true.

A false pattern can be formed, especially for tiny size structures or non-natural items like text –> loss function issue; some physical effects in the image, such as shadow, will appear artificial.

## 5.6 White-Box-Cartoonization

We suggest identifying three white-box representations from photos separately:

1. The portrayal on the surface
2. The portrayal of the structure

3. The depiction of texture

In the celluloid style process, the surface representation refers to the sparse colour blocks and flattens global content, whereas the structural representation refers to the smooth surface of cartoon pictures.

The texture representation in cartoon pictures reflects high-frequency texture, curves, and details. A Generative Adversarial Network (GAN) architecture is utilised to learn the extracted representations and cartoonize photos.

### 5.6.1 Surface Representation

- The surface representation mimics the cartoon painting approach, in which painters use coarse brushes to sketch rough drawings with smooth surfaces that resemble comic pictures.
- For edge-preserving filtering, a differentiable guided filter is used to smooth pictures while maintaining the global semantic structure.
- Edge-preserving filtering is a type of image processing that removes noise and textures while keeping sharp edges. The median, bilateral, directed, and anisotropic diffusion filters are examples.

**Surface Loss Formula**

Note that a discriminator Ds is used to determine if model outputs and reference cartoon pictures have comparable surfaces, and a guide generator G is used to learn the information contained in the extracted surface representation.

## 5.7 Structure Representation

To segment photos into discrete regions, we initially utilised the felzenszwalb method. We use selective search to combine segmented regions and obtain a sparse segmentation map since superpixel techniques only examine pixel similarity and neglect semantic information.

Standard superpixel algorithms use an average of the pixel value to colour each separated region. By analysing the processed dataset, we discovered that this decreases global contrast,

$$L_{surface}(G, D_s) = \log D_s(F_{dgf}(I_c, I_c)) + \log(1 - D_s(F_{dgf}(G(I_p), G(I_p))))$$

Where,

G = Generator,

Ds = Discriminator,

Ic = Reference Cartoon Image,

Ip = Input Photo,

Fdgf = It takes an image I as input and itself as a guide map, returns extracted surface representation Fdgf (I, I) with textures and details removed.

**Figure 5.7: Surface Loss Formula Equation**

darkens pictures, and generates a hazing effect on the final output. As a result, we present an adaptive coloration algorithm.

Adaptive coloring formula,

$$S_{i,j} = (\theta_1 * \bar{S} + \theta_2 * \check{S})^{\mu}$$

$$(\theta_1, \theta_2) = \begin{cases} (0, 1) & \sigma(S) < \gamma_1, \\ (0.5, 0.5) & \gamma_1 < \sigma(S) < \gamma_2, \\ (1, 0) & \gamma_2 < \sigma(S). \end{cases}$$

Where we find $\gamma_1 = 20$, $\gamma_2 = 40$ and $\mu = 1.2$ generate good results.

**Figure 5.8: Adaptive Coloring Equation**

This effectively enhances the contrast of images and reduces the hazing effect.

(a) and (b) show segmentation maps with different colouring methods, while (c) and (d) shows results generated with different colouring methods. Adaptive colouring generates results that are brighter and free from hazing effects.

### 5.7.1 Structure Loss Formula

Note: To enforce spatial restrictions between our results and the retrieved structure representation, we leverage high-level features extracted by a pre-trained VGG16 network.

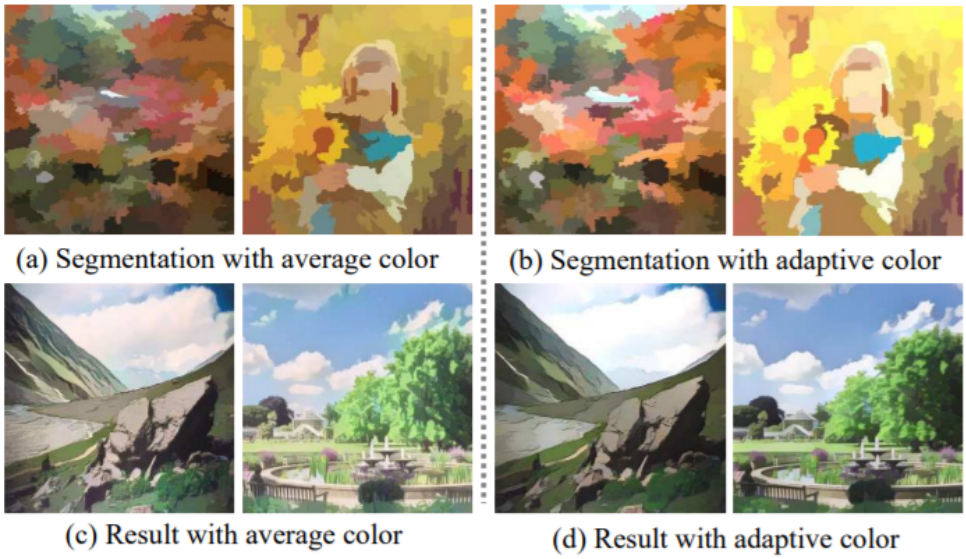


**Figure 5.9: Adaptive Coloring Algorithm**

$$L_{structure} = \| VGG_n(G(I_p)) - VGG_n(Fst(G(I_p))) \|$$

Where,

G = Generator,

Ip = Input Photo,

Fst = Structure Representation Extraction.

**Figure 5.10: Structure Loss Formula**

### 5.7.2 Texture Representation

The high-frequency elements of cartoon pictures are important learning objectives, but brightness and colour information help differentiate cartoon images from real-life photos. As a result, we suggest a random colour shift method. With luminance and colour information removed, the random colour shift may create random intensity maps.

Frcs converts colour pictures into single-channel texture representations, preserving high-frequency textures while reducing the effect of colour and brightness.

$$L_{texture}(G, D_t) = \log D_t(F_{rcs}(I_c)) + \log(1 - D_t(F_{rcs}(G(I_p))))$$

Where,

G = Generator,

Dt = Discriminator,

Ic = Reference Cartoon Image,

Ip = Input Photo,

Frcs = Extract single-channel texture representation from colour images, which retains high-frequency textures and decreases the influence of colour and luminance.

**Figure 5.11: Textual Representration Formula**

### 5.7.3 Textural Representation Formula

### 5.7.4 Agile Methodology

Agile methodology is a development and testing strategy that encourages continuous iteration throughout the project's software development life cycle. Unlike the Waterfall paradigm, the Agile model allows both development and testing to take place at the same time. It is a new method to software development that incorporates iterative development and incremental product delivery. Agile technique is dynamic, context-specific, change-aggressive, and growth-oriented.

Four basic values are emphasised in agile software development.

1. Interactions between individuals and teams over procedures and tools
2. Useful software over extensive documentation
3. Customer involvement in contract negotiations
4. Responding to change in accordance with a plan

### 5.7.5 System's Operation

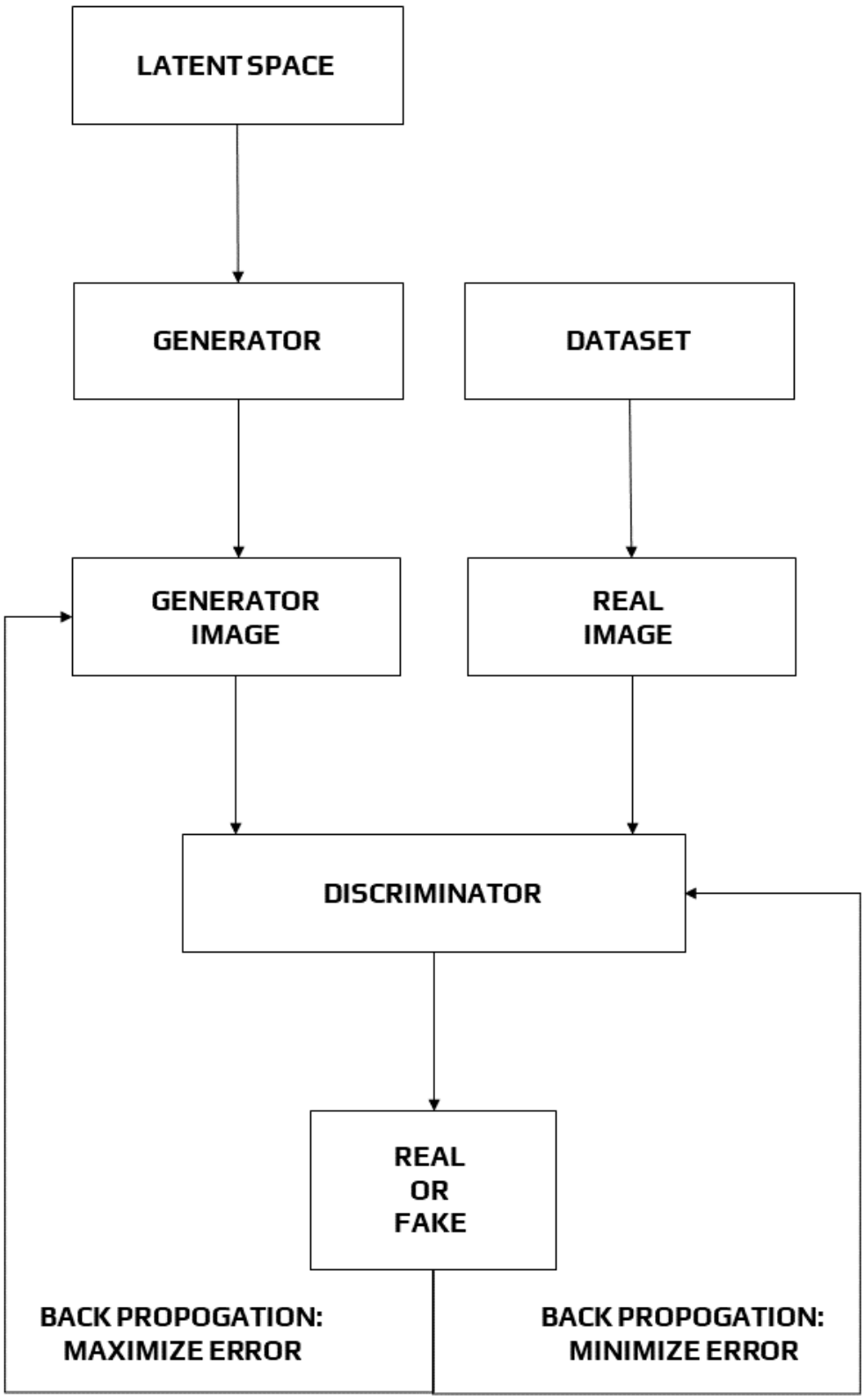


**Figure 5.12: Working of the system**

# Chapter 6

# Performance Evaluation

## 6.1 Previous Methods

Our technique is compared to four algorithms that represent:

1. Neural Style Transfer
2. Image-to-Image Translation
3. Image Abstraction
4. Image Cartoonization

## 6.2 Evaluation Metrics

We give findings from qualitative trials with information from four distinct approaches and original photographs, as well as qualitative analysis. In quantitative trials, we employ Frechet Inception Distance (FID) to assess performance by computing the distance between the source and target picture distributions. Candidates in the user research are asked to score the results of several approaches on a scale of 1 to 5 in cartoon quality and overall quality. Higher scores indicate higher quality.

## 6.3 Validation of Cartoon Representations

A categorization experiment and a quantitative experiment based on FID are done to confirm our suggested cartoon representations as acceptable and effective. On our training dataset, we

train a binary classifier to discriminate between real-world photographs and cartoon ones.

In our architecture, the classifier is created by adding a fully linked layer to the discriminator. The trained classifier is next tested on the validation set to ensure that each cartoon representation has an impact.

We discovered that the derived cartoon representations successfully fooled the trained classifier, since it obtains poorer accuracy in all three extracted cartoon representations when compared to the original pictures.

The estimated FID metrics corroborate our hypothesis that cartoon representations assist bridge the gap between real-world pictures and cartoon images, since all three extracted cartoon representations had lower FID than the original images.

## 6.4 Qualitative Comparison

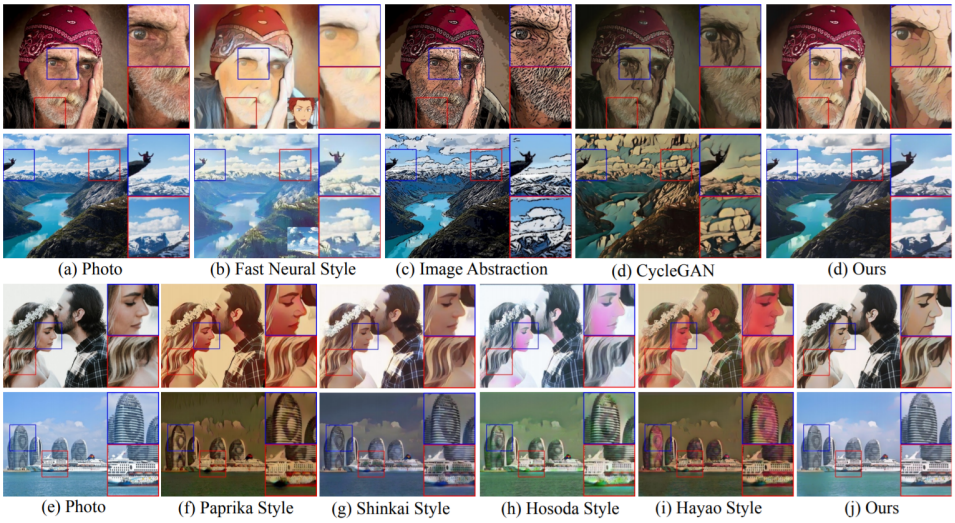


**Figure 6.1: Qualitative comparison**

Comparisons between our method and previous methods are shown in Figure 6.1

The white-box framework aids in the creation of crisp outlines. Image abstraction results in noisy and jumbled contours, while earlier approaches fail to produce clean contours.Our technique, on the other hand, has distinct borders, such as a human face and clouds. Color harmony is often helped by cartoon depictions. CycleGAN creates Fast Neural Style results in overly smoothed colour and darkened visuals. CartoonGAN distorts colours in the form of human features and ships.Our technique, on the other hand, prevents erroneous colour changes to things like faces and ships.

Finally, although our technique efficiently minimises artefacts while keeping small details, such as the guy sitting on the stone, all other methods result in over-smoothed or distorted features. High-frequency artefacts are also caused by approaches like CycleGAN, picture abstraction, and various CartoonGAN styles. To summarise, our technique exceeds prior methods in terms of producing photos with harmonious colour, crisp borders, fine details, and little noise.

## 6.5 Quantitative Evaluation

The Frechet Inception Distance (FID) is a commonly used metric for assessing the quality of synthetic pictures. To extract high-level features from photos and calculate the distance between two image distributions, the pre-trained Inception-V3 model is employed. FID is used to assess the performance of prior approaches as well as our own. Our approach produces photos with the lowest FID to cartoon image distribution, indicating that it produces the most cartoon-like outcomes.

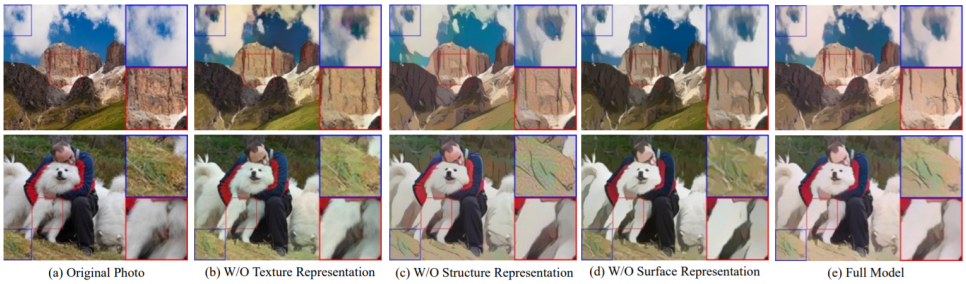


**Figure 6.2: Ablation study by removing each component**

The grassland and the dog's leg have uneven textures,seen in Figure 6.2(a). This is due to lack of high-frequency information preserved in the surface representation, which reduces the model's capacity to cartoonize. Figure 6.2(b) shows high-frequency sounds caused by ablating the structural representation. On the meadow and the mountain, there is a lot of pepper and salt. Because of the structural representation, pictures were flattened and high-frequency information was lost.

Figure 6.2(c) shows cloud borders that are hazy. This is the case because directed filtering decreases high-frequency information while keeping smooth surfaces. The results of our whole model are shown in Figure 6.2(d), which include smooth features, clear boundaries, and dramatically decreased noise. Finally, these three representations help to improve the cartoonization capability of our technique.

# Chapter 7

# Conclusion

## 7.1 Conclusion

In this study, we provide a GAN-based deployed white-box controlled image cartoonization system that can produce high-quality cartoonized pictures from real-world photos.

The surface representation, the structure representation, and the texture representation are the three cartoon representations of an image. Three representations for network training are extracted using image processing modules that correspond to each other.

To validate the performance of our technique, we conducted several quantitative and qualitative tests. Ablation experiments are also carried out to show how each representation affects the brain.

## 7.2 Future Prospects

Furthermore, extensive study and development in this white box cartoonization might lead to a wide range of applications:

- Quickly creates prototypes or sprites for anime, cartoons, and games.
- It may be used to create simple art since it subdues face characteristics and information in general.
- Games can simply input shortcut scenes without needing motion capture, and it may be modelled as an assistance to graphic designers or animators.

# Bibliography


[1] J. Bruna, P. Sprechmann, and Y. LeCun. Super-resolution with deep convolutional sufficient statistics. In *International Conference on Learning Representations (ICLR)*, 2016

[2] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi IEEE *Transactions on Pattern Analysis and Machine Intelligence*, IEEE Transactions on Pattern Analysis and Machine Intelligence

[3] Yang Chen, Yu-Kun Lai, and Yong-Jin Liu. Cartoongan: Generative adversarial networks for photo cartoonization. In IEEE Conference on Computer Vision and Pattern Recognition, 2018.

[4] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. *Perceptual losses for real-time style transfer and super-resolution*. In European Conference on Computer Vision, Springer, 2016.

[5] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. *Image-to-image translation with conditional adversarial networks*. In IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[6] Eirikur Agustsson and Radu Timofte. challenge on single image super-resolution: Dataset and study. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, July 2017

[7] Hussein A Aly and Eric Dubois. Image up-sampling using total-variation regularization with a new observation model. *IEEE Transactions on Image Processing*

[8] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. *arXiv* preprint *arXiv*

[9] Qingnan Fan, Jiaolong Yang, David Wipf, Baoquan Chen, and Xin Tong. Image smoothing via unsupervised learning

[10] Zeev Farbman, Raanan Fattal, Dani Lischinski, and Richard Szeliski. Edge-preserving decompositions for multi-scale tone and detail manipulation.

[11] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*

[12] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *European Conference on Computer Vision*

[13] P. L. Rosin and J. Collomosse. Image and Video-Based Artis- tic Stylisation. Springer, 2013.

[14] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. CoRR

[15] L. Gatys, A. Ecker, and M. Bethge. Image style transfer using convolutional neural networks *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*

[16] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition, 2019*

[17] Jan Eric Kyprianidis and J¨urgen D¨ollner. Image abstraction by structure adaptive filtering.

[18] Li Xu, Jimmy Ren, Qiong Yan, Renjie Liao, and Jiaya Jia. Deep edge-aware filters. In *International Conference on Machine Learning,2015*

[19] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycleconsistent adversarial networks. In Proceedings of *IEEE International Conference on Computer Vision, 2017*

[20] JueWang, Yingqing Xu, Heung-Yeung Shum, and Michael F Cohen. Video tooning. In *ACM Transactions on Graphics, 2004*