CartoonifyGAN:Generative Adversarial Network for Image Cartoonization

A PROJECT REPORT

Submitted by

SIJI JOSE(TKM20MCA-2037)

to

The APJ Abdul Kalam Technological University

In partial fulfillment of the requirements for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

Thangal Kunju Musaliar College of Engineering
Kerala

DEPARTMENT OF COMPUTER APPLICATIONS

JULY 2022

DECLARATION

I undersigned hereby declare that the project report on "**CartoonifyGAN: Generative Adversarial Network for Image Cartoonization**" , submitted for partial fulfillment of the requirements for the award Of degree of Master of Computer Applications of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under supervision of **Prof. Jasmin M R**.This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources.I also declare that we have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. I understand that any violation of the abovewill be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from Whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree ,diploma or similar title of any other University.

Kollam

12/07/2022

SIJI JOSE

DEPARTMENT OF COMPUTER APPLICATIONS

TKM COLLEGE OF ENGINEERING

CERTIFICATE

This is to certify that, the project report entitled **"CartoonifyGAN: Generative Adversarial Network for Image Cartoonization"**submitted by **SIJI JOSE(TKM20MCA-2037)**to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the degree of Master of Computer Applications, is a bonafide record of the project work carried out by her under our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

Internal Supervisor	Head of the Department	External Examiner

ACKNOWLEDGEMENT

First and foremost I thank GOD almighty and my parents for the success of this project.I owe sincere gratitude and heart full thanks to everyone who shared their precious time and knowledge for the successful completion of my project.

I am extremely grateful to **Dr.Fousia.M.Shamsudeen**,Head of the Department,for providing me with best facilities.

I would like to thank my project guide **Prof.Jasmin M R**, Department of Computer Applications ,who motivated me throughout the work of my project.

I profusely thank all other faculty members in the department and all other members of TKM College of Engineering, for their guidance and inspirations throughout my course of study.

I owe my thanks to my friends and all others who have directly or indirectly helped me in the successful completion of this project.

SIJI JOSE

ABSTRACT

A solution for converting real-world pictures into cartoonified pictures that is both useful and exciting in computer vision is proposed. Our method is organised as a knowledge-based plan that has recently gained popularity as a method of stylizing images in creative forms such as painting. Existing artistic style methods on the other hand do not produce satisfactory results because (1) cartoon styles have distinct features such as elite resolution and generalizability (2) cartoon images have smooth edges with obvious color changes. In this project, it provide cartoonifyGAN , a Generative Adversarial Network (GAN) methodology for cartoonization. It utilize mismatched photographs and hilarious images for teaching cartoonization which is a simple process and makes excellent cartoon drawings from real-world pictures.

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Definition
GAN	Generative Adversarial Network
FID	Frechet Inception Distance

Chapter 1

Introduction

Cartoons are a popular type of art that we see on a daily basis. In addition to aesthetic interests, there are uses ranging from print media publication to narrative for children's education. Many iconic cartoon images, like other types of art, were inspired by true-life events. Manually reproducing real-world circumstances in cartoon styles, on the other hand, takes effort and a great level of ingenuity. Every line and colour section in the target scenes must be drawn by the cartoonists in order to produce high-quality cartoons. On the other hand, conventionally capable picture altering tools and algorithms now in use are unable to deliver sufficient cartoonization outcomes. As a result, technologies designed expressly to automatically translate real-world images into high-quality cartoon-style visuals are quite useful, and artists can save a significant amount of time. CartoonifyGAN, a GAN-based technique for image cartoonization, is proposed in this project. For training, a set of still photographs and a set of cartoon images were used. While keeping training data collection simple, outstanding results do not need matching or correspondence between two sets of images. From the perspective of computer vision algorithms, cartoon stylization seeks to shift images from the photo manifold into the cartoon manifold while keeping the content. The VGG network is used for content loss because it has enough flexibility to reproduce smooth shading. This suggests employing two straightforward yet efficient loss functions in conjunction with a specialized GAN-based architecture to accomplish the goal.

1.1 Problem Statement

Cartoons are a popular type of art that we see daily. Their applications range from print media publication to a narrative for children's education, in addition to aesthetic interests. Many iconic cartoon images, like other types of art, were inspired by true-life events. On the other hand, it takes patience and a great level of inventiveness to manually recreate real-world scenarios in cartoon forms. Cartoonists must meticulously draw each line and shade each area of the target scenes in order to produce high-quality cartoons. On the other hand, typical image editing software and algorithms in use today are unable to deliver sufficient cartoonization results.As a result, systems designed especially for this purpose are highly helpful, and artists may save a large amount of time. Real-world images could be automatically converted by these techniques into stunning cartoon-style graphics. CartoonifyGAN, a GAN-based technique for cartoonizing images, is proposed to eliminate this. For educational purposes, it uses both a series of still photos and a set of cartoon graphics. There is no requirement for matching or correspondence between two sets of photographs in order to achieve high-quality outcomes while creating training data simple to collect. Cartoon stylization fulfils the purpose of converting images from the while retaining the integrity of the photo manifold of the information from the viewpoint of computer vision algorithms. The VGG network, which offers enough flexibility for replicating smooth shading, is employed for content loss. This suggests achieving the goal by combining a specific GAN-based design with two straightforward yet efficient loss functions.

1.2 Objectives

- The main goal is to create a cartoonization architecture using a compact generative adversarial network (GAN).
- Pre-processing is carried out to raise the standard of the training photos.
- Generator has been trained to create fake images that resemble real images.
- In order to discriminate between authentic and false photos, a discriminator is trained.
- VGG16 is used to extract image features for content loss,which provides sufficient flexibility for reproducing smooth shading..
- After training for several epochs, the Generator model is saved and this model is used for testing.

Chapter 2

Literature Survey

A thorough investigation and analysis of the literature on a particular subject constitutes a literature review. When using a literature review, research questions are identified, and then answers are sought by searching for and analyzing relevant literature. The importance of literature reviews stems from the fact that new insights can be developed by re-analyzing the study's findings. A literature review summarizes and explains the whole and current state of knowledge on a subject as found in scholarly publications and journal articles. University students may be required to write one of two different sorts of literature reviews: one as a stand-alone project for a course, and the other as an introduction to or preparation for a bigger work, usually a thesis or research report. Depending on the type of review you are writing, your topic, point of view, and type of hypothesis or thesis argument will all be influenced. Reading published literature reviews or the introductory sections of theses and dissertations in your field of study will help you better comprehend the distinctions between these two forms. Examine the manner they address the issues and the organization of their arguments.

2.1 Purpose of the Literature Review

1. By choosing top-notch articles or studies that are pertinent, significant, vital, and valid and compiling them into one comprehensive report, it makes it simple for readers to conduct research on a given issue.
2. By requiring them to describe, assess, and compare original research in that particular field, it gives researchers who are starting their work in a new area a great place to start.
3. It guarantees that researchers don't redo work that has previously been completed.
4. It can offer hints as to where future research is going or suggest areas on which to concentrate.

5. It emphasizes the essential findings.
6. 6. It points out gaps, discrepancies, and inconsistencies in the literature.
7. It offers a constructive critique of the methods and strategies used by other researchers.

2.2 Related works

2.2.1 Y. Jing et al., "**Stroke controllable fast style transfer with adaptive receptive fields**," in Proc. Eur. Conf. Comput. Vis., 2018, pp. 238–254.

METHOD USED : Non-Photorealistic Rendering

Several NPR algorithms have been developed to automatically or partially recreate specific visual styles, such as cartoons. A cartoonish appearance is achieved in certain works by representing 3D forms with simple shading. Games, animated films, and movies have all made use of the technique known as cel shading, which may help artists save a lot of time. It is considerably more challenging to turn existing pictures or movies into cartoons, as was the case with the topic of this study. Many techniques have been devised to produce flat-shaded graphics that resemble cartoon styles.. Examples of such tactics include image filtering or formulations in optimization issues. Image filtering is used by Winnemölleret al. to create a cartoon version of a given image. On the other hand, complex aesthetic styles cannot be represented by simple mathematical techniques. For instance, the high level of abstraction that an artist would create, such as clearly delineating object borders, cannot be achieved by applying uniform optimization or filtering to the entire image. Alternative methods rely on image/video segmentation to improve results, although this sacrifices some user interaction. There are also specific algorithms for portraiture, where semantic segmentation is automatically constructed by locating face components. But standard images cannot be handled by such techniques.

2.2.2 Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song, “**Neural style transfer: A review**,” IEEE Trans. Vis. Comput. Graphics, vol. 26, no. 11,pp. 3365–3385, Nov. 2020.

METHOD USED :Neural Style-Transfer Method

A number of people are interested in using convolutional neural networks (CNNs) to overcome various Machine vision challenges. As an alternative to developing special NPR algorithms that require a great deal of work for each style, style transfer has been actively investigated. In contrast to traditional style transfer methodologies that that require paired style/non-style photos, new research has shown the VGG network's strong capacity to extract semantic features of objects, which are critical in stylization. The VGG network was trained for object recognition. Consequently, a more effective style transfer approaches that do not need matched training images have been created. The domains of computer vision and image processing typically use convolutional neural networks (CNN) to address problems. Contrary to traditional style transfer techniques, which demand both style and non-style photos, semantic properties of objects can be carried out more effectively by the VGG network trained for object recognition., and it is one of the fundamental aspects of stylization. Image to Image translation is a different format that deals with moving images from one domain to another. It enhances the quality of photographs by turning photos into paintings, cartoons, and sketches. Within a few days, bi-directional translation models are suggested. Zhu et al. convert Rain into Winter and an unpaired drawing into a painting images.

2.2.3 I. Goodfellowet al., “**Generative adversarial nets**,” in Proc. Adv. Neural Inf. Process. Syst., 2014, pp. 2672–2680.

METHOD USED :GAN-Based Methods

GAN has produced cutting-edge achievements in image in painting, text to image translation, image super-resolution, and other domains. However, when undertaking image synthesis, generic GAN requires matching picture sets, which is frequently impractical because it is difficult to locate data sets that contain both actual and synthetic images.

(i) J. Chen, G. Liu, and X. Chen, “**AnimeGAN: A novel lightweight GAN for photo animation**,” in Proc. Int. Symp. Intell.Comput. Appl., 2019, pp. 242–256.

METHOD USED :AnimeGAN

To create excellent anime-faces, a new GAN-based translator dubbed AnimeGAN was developed. A new generator architecture is specifically presented to maintain the overall structure of the original photo-face while simultaneously transferring color/texture styles and altering local facial forms into counted portions that correspond to reference anime-styled faces. A large computational cost and a 77.3 FID score are present here.

(ii) J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "**Unpaired image-to-image translation using cycle-consistent adversarial networks,**" in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 2242–2251.

METHOD USED :CycleGAN

CycleGAN uses a cyclical training approach. With the use of unpaired training data, it may produce striking creative representations. CycleGAN, which obtains a FID score of 91.95, which can only study one artist at a time's style, requires high computational costs to train several pairs of CycleGAN models. It separates the generator's encoder and decoder, reusing them to address for the mentioned issues..

(iii) Y. Chen, Y.-K. Lai, and Y.-J. Liu, "**CartoonGAN: Generative adversarial networks for photocartoonization,**" in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 9465–9474.

METHOD USED :CartoonGAN

Using real-world photographs as the starting point, CartoonGAN creates high-quality cartoon-style visuals. It first suggests a new adversarial loss that promotes edges for transparent edges, with the aim of faithfully replicating the properties of cartoon graphics. Additionally, it suggests a straightforward yet effective initialization phase to aid in improving convergence. The studies demonstrate that CartoonGAN can learn a model with a FID score of 58.71 that can efficiently and with excellent quality convert photographs of real-world scenes into cartoon-style graphics. Even so, it reveals a few minor discrepancies in the local information.

(iv) Yongsheng Dong , Wei Tan, Dacheng Tao , LintaoZheng , and Xuelong Li , "**CartoonLossGAN: Learning Surface and Coloring of Images for Cartoonization**" in IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 31, 2022 485

METHOD USED :CartoonlossGAN

A generative adversarial network framework called CartoonLossGAN creates colourful cartoon illustrations by studying the surface and colouring of images to mimic the process of first sketching and then colouring cartoons. The suggested cartoon loss function can simulate both the drawing and colouring procedures to learn the cartoon's smooth surface. In order to create a small, generative adversarial network (GAN) architecture, it also make use of the discriminator's encoder portion without employing a cyclic method. Numerous experimental findings show that CartoonLossGAN, with a FID score of 53.95, can produce amazing cartoon-style visuals.

Generative Adversarial Network (GANs) discussed above helps to create cartoon-like images which is not obivious. The proposed CartoonifyGAN aims to produce visuals that resemble cartoons in every way. With addition, it recommends a quick and easy setup procedure to aid in convergence. Tests have shown that a model that accurately and efficiently transforms photographs of realistic scenes into cartoons may be learned by CartoonifyGAN and a reduction in FID Score.

Chapter3

Methodology

Cartoonization is a unique form of artistic style transfer that is also challenging to implement in image processing. Due to the fact that cartoon-style graphics typically have smooth surfaces and sharp edges whereas artistic style images typically include delicate strokes and rich hierarchical colour shifts, current artistic style transfer techniques cannot produce cartoon-style images that are acceptable..A cartoonified generative adversarial network (CartoonifyGAN) is suggested as a solution to this problem. Traditional cartoons are created by manually drawing from the real world. Images are preprocessed to balance the imbalanced data. Through the competitive training of the generator and discriminator, GAN is utilised to produce images that are realistic. The implementation of a pre-trained CNN model (VGG16) to extract features.

3.1 Proposed System

Cartoonization is a unique form of artistic style transfer that is also challenging to implement in image processing. Because cartoon-style images have smooth surfaces without evident colour changes,sharp edges, and artistic style images frequently feature delicate strokes and rich hierarchical colour changes, current artistic style transfer methods cannot produce appropriate cartoon-style images. A cartoonified generative adversarial network (CartoonifyGAN) is suggested as a solution to this problem. Traditional cartoons are created by manually drawing from the real world. Images are preprocessed to balance the imbalanced data. Through the competitive training of the generator and discriminator, GAN is utilised to produce images that are realistic. The implementation of a pre-trained CNN model VGG16 to extract features Our goal is to solve the minimum-maximum problem ,using

$(G^*, D^*) = \arg\min_G \max_D L(G, D)$

3.2 System Architecture

In the preprocessing step, two datasets, such as cartoons and real-world photos, are preprocessed with a variety of smoothing techniques before being used in the training phase as shown in Figure.3.1. This prepares the datasets for use in the training phase. After that, the encoder of the discriminator will convert it, and then it will be sent to the residual blocks and convolutional layers of the generator to undergo additional processing there. Following processing, the data that is fed into deep neural networks leads those networks to generate images that are consistent with the data that was supplied to them. Application of the trained VGG-16 model to the newly produced photographs in order to extract attributes from them. The discriminator is what is used to determine the adversarial loss based on the input that was produced, and it does this using the information that was provided. The generator model has been saved after going through a certain number of training epochs, and it is now getting ready to be made available to the general public. The generator creates cartoon-like representations of the real-world input during the generating phase as shown in Figure.3.2. The input picture is altered using a few preprocessing stages, and then it is fed through the CartoonifyGAN to produce the desired outcomes.

3.2.1 Dataset

The AnimeGAN cartoon dataset is utilised. The test data contains solely real-world images, whereas the training data includes both, and all training data is processed to 256-bit precision.

- Real-World Photos: 790 photographs are used for testing and 6656 photos from the actual world are used for training. All of the images are processed at 256 × 256. 45 high-resolution photos are also used to display the outcomes of our model as cartoons.
- Cartoon Images: Smooth, colour, and grayscale cartoon images are all types of cartoon images. Cartoons are only used for training purposes.

Figure 3.1 : Training Phase

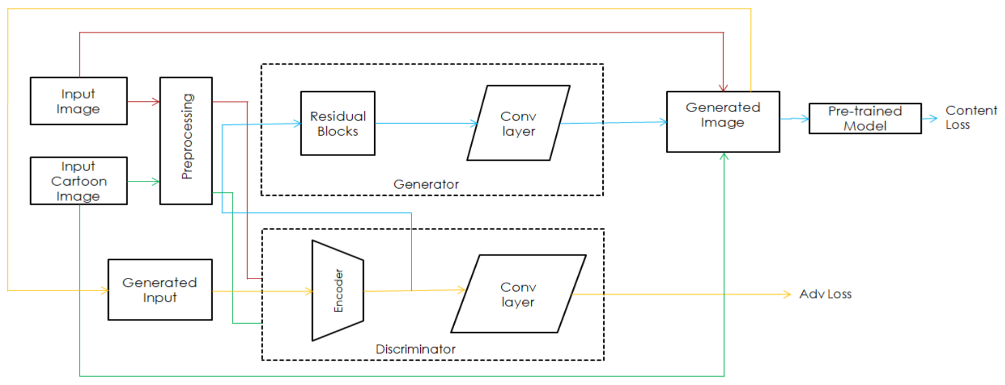
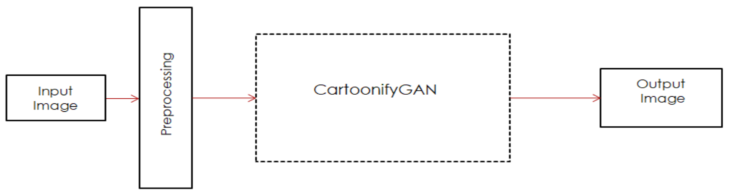


Figure 3.2 : Generating Phase



3.2.2 Data Preprocessing

It converts each cartoon image to 256 × 256 first. Next, apply smoothing techniques to the cartoon images' edges. Then, fundamental morphological work like erosion is completed. Here, erosion is used to separate two related objects and remove faint white sounds. It reduces the features of an image and erodes the foreground object's edges.

3.2.2.1 Smoothing Techniques

Data smoothing is predicated on the idea that one is measuring a variable that is both slowly changing and tainted by random noise. In addition to the smoothing effect that filtering has on photographs, the Gaussian kernel is another representative method that has been used. The weights contained within the kernel are used to determine the weighted average of neighbouring points (pixels) in a picture, have the shape of the function "Gaussian distribution," as its name implies.

3.2.2.2 Erosion

Erosion reduces the image pixels, or it uses element B to make element A smaller. Pixels on object boundaries are removed. The output pixel's value is the lowest of all the pixels in its immediate vicinity. A pixel is set to 0 if any adjacent pixels have the value 0.

3.2.3 Training the model

In the preprocessing step, two datasets, such as cartoons and real-world photos, are preprocessed with a variety of smoothing techniques before being used in the training phase as shown in Figure.3.3. This prepares the datasets for use in the training phase. After that, the encoder of the discriminator will convert it, and then it will be sent to the residual blocks and convolutional layers of the generator to undergo additional processing there. Following processing, the data that is fed into deep neural networks leads those networks to generate images that are consistent with the data that was supplied to them. .
Application of the trained VGG-16 mode to the newly produced photographs in order to extract attributes from them. The discriminator is what is used to determine the adversarial loss in Figure.3.4 ,based on the input that was produced, and it does this using the information that was provided. The generator model has been saved after going through a certain number of training epochs, and it is now getting ready to be made available to the general public.

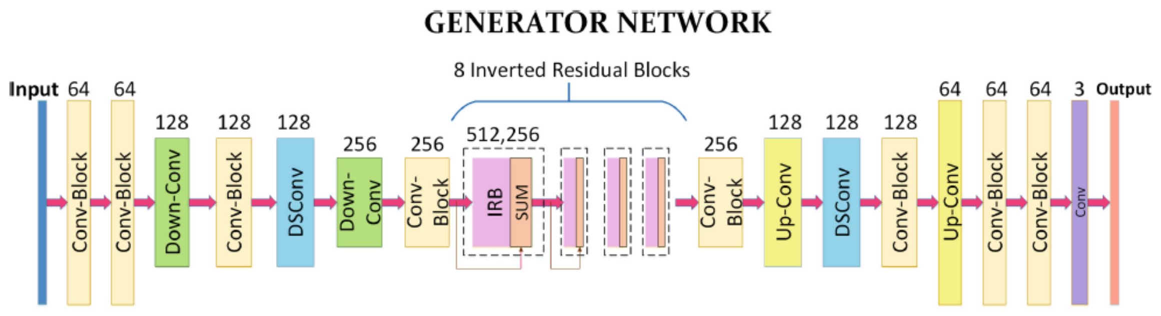


Figure 3.3

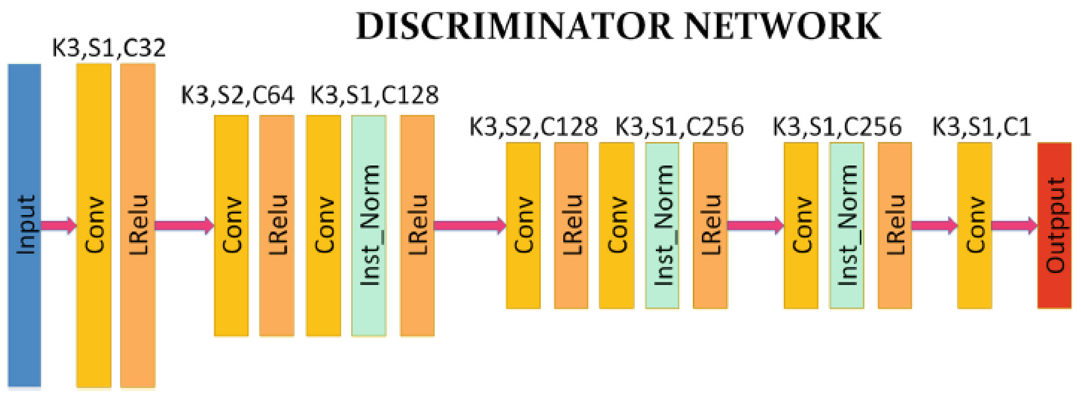


Figure 3.4

3.2.4 Testing the model

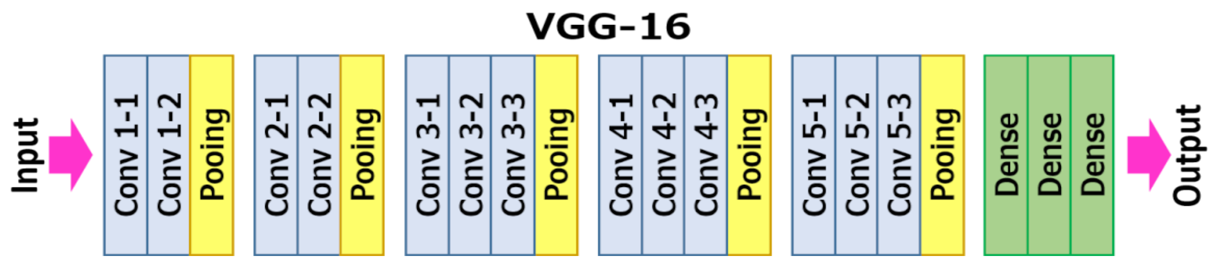
The generator creates cartoon-like representations of the real-world input during the generating phase as shown in Figure.3.2. The input picture is altered using a few preprocessing stages, and then it is fed through the CartoonifyGAN to produce the desired outcomes as in Figure.4.4.

3.2.5 Pre-trained VGG -16 Architecture

In the content loss procedure, the generated image's characteristics are compared to the content image. It must be ensured that the output image preserves the same content even while the algorithm modifies the style. In this way, the style elements are applied from the style image while maintaining the authenticity of the content image. First, two instances of the same pre-trained VGG-16 image categorization are employed.

CNNs are used with loss networks. These networks will receive a test picture (created) and a reference image (content), respectively. The outputs from these two classifiers are used as inputs for the loss function. The content loss is retrieved with the help of pre-trained VGG16.

Figure 3.5 : VGG-16 Architecture



3.3 Software requirements and specification

The software used for the project:

• Python

• Google Colaboratory

• Django Framework

1. PYTHON

In 1989, Guido Rossum developed the object-oriented programming language Python. It is ideal made for the quick prototyping of complex applications. It is extensible to C or C++ and offers interfaces to a number of OS system calls and libraries. Numerous large organisations, including Google, YouTube, BitTorrent, and NASA, employ the Python programming language.. Neural networks, artificial intelligence, and other cutting-edge computer science fields all make extensive use of the Python programming language. Guido van Rossum designed Python, a high-level, open-source programming language, in the late 1980s and is currently run by the Python Software Foundation. It originated with the

ABC language, which he co-created early in his career. Python is a strong language that can be used to create GUIs, create games, and create web applications. It is a sophisticated language. In Python, reading and writing statements differ greatly from reading and writing statements in standard English. Python programs got to be processed before machines Because of this, they are not written in a language that computers can understand. Instead, Python programs have to be created before machine can operate them. The language of Python is easily understood. This suggests that each time a programme is executed , its interpreter reads the program's code and translates it into byte code that can be read by a computer. Python is an object-oriented language that lets users create and execute programmes by managing objects or data structures. In reality, Python has top-notch everything. In Python, every object, data type, function, method, and class has an equal standing.

Programming languages are developed to meet the needs of users and programmers for an effective tool to construct programmes that have an impact on people's lives, way of life, economy, and society.

By boosting productivity, improving communication, and boosting power, they help improve life. When a language falls short of expectations, it dies and is replaced by a more powerful language, which makes it obsolete. Python is an artificial language that has withstood the test of time and has stayed relevant across corporations, industries, programmers, and individual users. It is a vibrant, incredibly helpful language that is strongly advised as a primary programming language for individuals who want to get started with and learn about programming.

- Python is interpreted: Each time the interpreter goes through Python. It is not necessary to compile your software before running it. This is similar to PHP and PERL.
- Python is interactive: When writing programs in Python, you can sit at a prompt and immediately communicate with the computer.
- Python is an object-oriented language: Python supports the Object-Oriented programming style

or technique, which encapsulates code within objects.

.

• Python is an Excellent First Programming Language: Python is an excellent first programming language that allows for the development of a wide range of programmes, from simple text processing to web browsers to games.

• A simple and clear language that is just as effective as that of the main rival

• Its development is open source, allowing anyone to participate.

• Code that is as easy to understand as simple English

2. Google Colaboratory

Google created Google Colab to give anyone who requires access to GPUs and TPUs for building a machine learning or deep learning model free access to them. A more advanced version of Jupyter Notebook is Google Colab. Jupyter Notebook is a tool that allows you to edit and run Notebook documents using a web browser or an Integrated Development Environment (IDE). It will work with Notebooks instead of files. Notebook pages can include executable lines of code in addition to text, photographs, figures, tables, graphs, equations, and other graphical data. Simply said, writing executable documents that are human-readable is possible with Notebook documents. A Notebook's building blocks are called cells. A notebook's whole contents are made up of cells. There are two categories:

- Textcell: A textcell can contain text, photos, links, and much more. You can alter the contents of a text cell by double-clicking it. The text cell supports the Markdown markup language.
- Code cell: A code cell contains the executable code.The run button to the left of a code cell allows you to run the cell's code. The results of a cell run are shown underneath the cell.
- The interesting features that each contemporary IDE offers are abundant in Google Colab, in addition to many others. Below is a list of some of the more fascinating aspects.
- Interactive tutorials for learning neural networks and machine learning.

- Create and run Python3 programmes without a local setup.
- Use the Notebook to run terminal commands.
- Import data from outside resources like Kaggle.
- Upload notebooks to Google Drive.
- Google Drive notebook imports.
- Free GPUs, TPUs, and cloud services.
- Integrate with TensorFlow, OpenCV, and PyTorch.
- Upload or publish straight from or to GitHub

3. Django Framework

Django is a model-template-views (MTV) web framework that runs on Python. It is maintained by the independent non-profit Django Software Foundation (DSF), which was established in the US. Django's main objective is to make the process of creating intricate, database-driven websites simpler. Don't repeat yourself, low coupling, less code, component reuse, and "pluggability" are all strongly emphasised in the framework. Everywhere, including settings, files, and data models, uses Python. Django also has a customizable, dynamically built administrative creation, read, update, and delete interface. Although the callable objects that produce HTTP answers are referred to as "views," the MVC architecture of the fundamental Django framework may be thought of. It includes a regular-expression-based URL dispatcher, an object-relational mapper (ORM) that acts as a conduit between relational databases and data models (supplied as Python classes), and a mechanism for controlling HTTP requests ("Controller"). The basic structure also includes the following:

- a compact, independent web server for testing and development.
- a system for serialising and validating forms that can convert between HTML forms and data that can be stored in databases.
- a template system that makes use of the object-oriented programming idea of inheritance; • a caching framework that can make use of a variety of cache techniques.
- support for middleware classes that can perform specialised tasks and obstruct the processing of requests at certain points in the process

- a mechanism for internal dispatching events that enables an application's parts to communicate with one another using pre-defined signals\
- a serialisation system for Django model instances capable of generating and reading XML or JSON representations of those instances.
- an approach for expanding the template engine's capabilities

Chapter 4

Result And Discussion

Before being employed in the training phase, two datasets, such as cartoons and real-world photographs, are preprocessed in the preprocessing step, as illustrated in Figure.1. The datasets are now ready to be used in the training phase. It will then be converted by the discriminator's encoder before being forwarded to the generator's residual blocks and convolutional layers for further processing. After processing, the information supplied to deep neural networks causes those networks to produce visuals that are consistent with the information they received as input. The freshly created images are subjected to the pre-trained VGG-16 model [6] in order to extract properties from them The discriminator is what uses the supplied information to calculate the adversarial loss depending on the input that was produced. After passing through a predetermined number of training 210 epochs, the generator model was saved, and it is currently getting ready to be made publicly accessible. During the generating phase, the generator produces cartoon-like representations of the real-world input, as seen in Figure 2. A few preprocessing steps are used to modify the input image before it is passed into the CartoonifyGAN to get the desired results.

In this study, it was suggested using a CartoonifyGAN , Generative Adversarial Network used to create cartoon-like graphics from real-world photos. In addition to the VGG-16 network's sparse regularisation of high-level feature maps for content loss, which enables smooth shading replication, an unique adversarial loss that encourages clean edges is also suggested. The ultimate goal is to create images that are completely cartoon-like. Additionally, it suggests a simple setup process that is rapid to help with convergence. Tests have demonstrated that CartoonifyGAN is capable of learning a model that effectively

and accurately turns photos of genuine scenes into cartoons, as seen in Figures 4.5 and 4.6 and a decrease in FID Score in Figure 4.1. In this regard, it is far better than previous "state-of-the-art" stylization systems.

4.1 Performance and Evaluation

Given the wide range of opinions on the subject and the difficulty of quantifying them, it is difficult to determine how to analyse the consequences of creative style transfer . In this study, our model is assessed using the Frechet Inception Distance (FID). Using a trained VGG-16 model, FID gathers visual characteristics and defines the distinction between two images. Distributions can be compared by using the extracted image attributes to see how similar the two image distributions are to one another. The FID score dispersion is worse, which means that the produced image's distribution is more similar to the distribution of the original, legitimate reference image. To put it another way, the final image is more like the original.

Cartoon and content photos are given separate FID ratings. FID scores generated are used mainly for comparison. As shown in Table 4.1 , in order to determine if the created pictures are able to maintain the content images' semantic information, the FID score is calculated.

Table 4.1: FID scores

Model	FID Score to content loss
CycleGAN	91.95
AnimeGAN	77.36
CartoonGAN	58.71
CartoonLossGAN	53.95
CartoonifyGAN	52.81

The graph below shows the FID scores of different GANs.

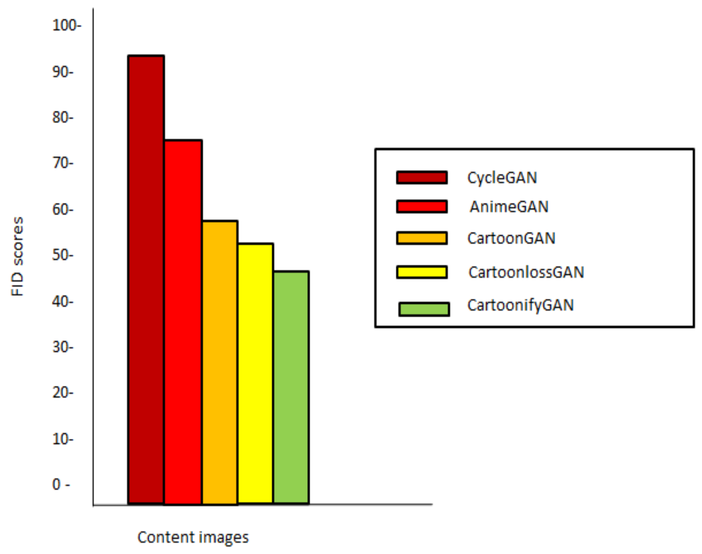


Figure 4.1 : GAN comparisons

- Content loss

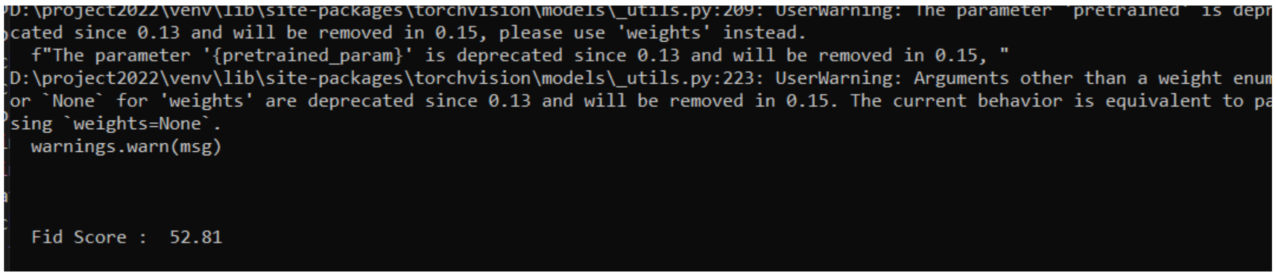
The generated image's features are contrasted with the content image in the content loss process. While the algorithm changes the style, it must be ensured that the output image retains the same content. In this manner, the authenticity of the content image is preserved, and the style elements are added from the style image. The identical pre-trained VGG-16 image categorization is first used twice. Loss networks are used with CNNs. A test image (created) and a reference image (content) will be supplied to these networks, respectively. These two classifiers' outputs are fed into the loss function as inputs.

- Adversarial loss

Adversarial Loss is the name of the loss that GANs employ. A constantly trained discriminator network determines the adversarial loss. It is a binary classifier that distinguishes between data from the real world and data generated by the generative network. It primarily takes the form of

$$\min_{G}\max_{D} V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

A discriminator neural network strives to discriminate between actual and generated samples, whereas the generator network attempts to trick the discriminator.



```
D:\project2022\venv\lib\site-packages\torchvision\models\_utils.py:209: UserWarning: The parameter 'pretrained' is depr
cated since 0.13 and will be removed in 0.15, please use 'weights' instead.
  f"The parameter '{pretrained_param}' is deprecated since 0.13 and will be removed in 0.15, "
D:\project2022\venv\lib\site-packages\torchvision\models\_utils.py:223: UserWarning: Arguments other than a weight enum
or `None` for 'weights' are deprecated since 0.13 and will be removed in 0.15. The current behavior is equivalent to pa
sing `weights=None`.
  warnings.warn(msg)



  Fid Score :  52.81
```

Figure4.2:FID score of CartoonifyGAN

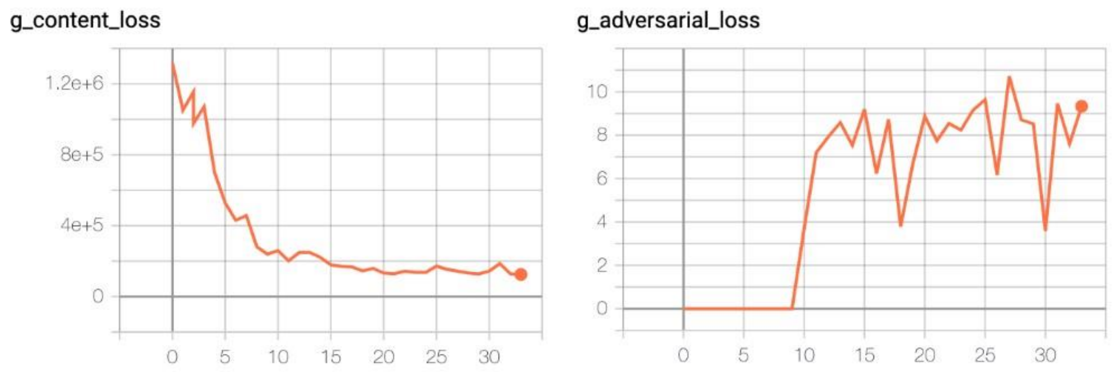


Figure4.3:Content loss, Adversarial loss

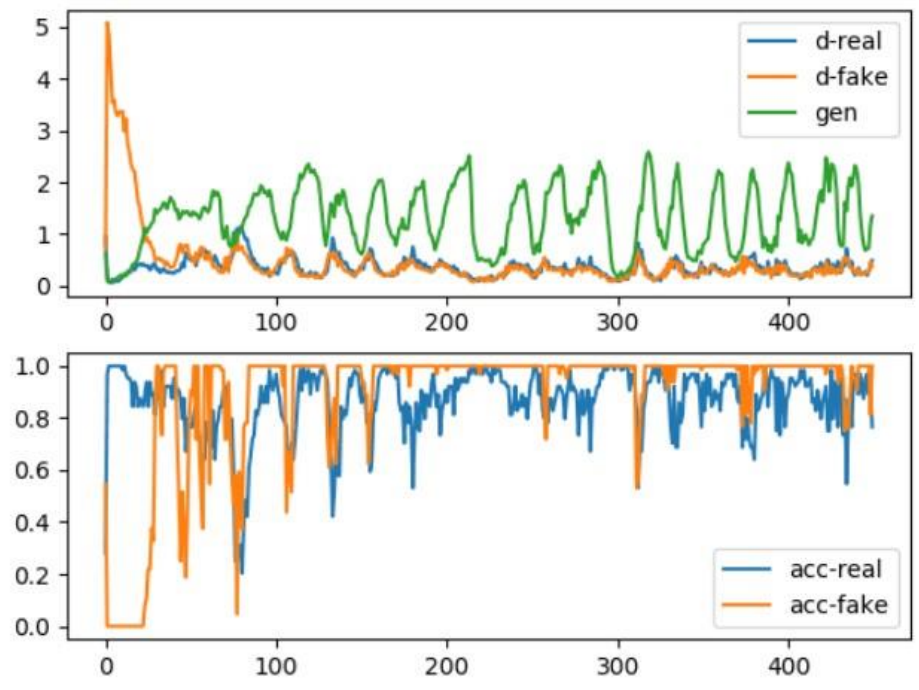


Figure 4.4:Accuracy-Loss Graph

The discriminator has an accuracy of 50% if the generator always succeeds, which is comparable to tossing a coin. This puts the convergence of the GAN as a whole in danger. If the generator keeps learning on these subpar training signals, the discriminator's feedback may lose meaning over succeeding epochs by producing outputs with equal probability.

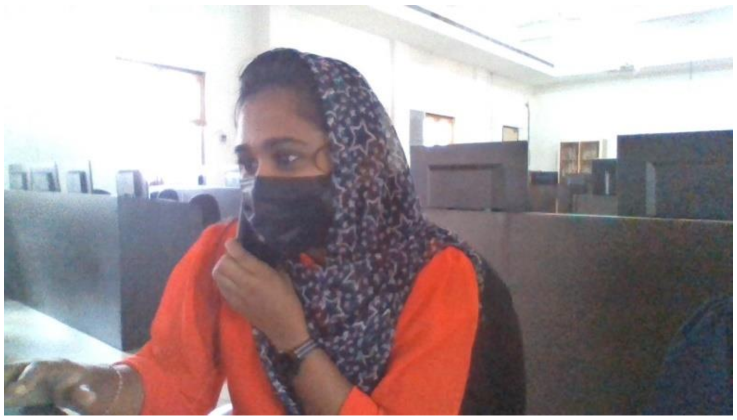


Figure4.5:Input image

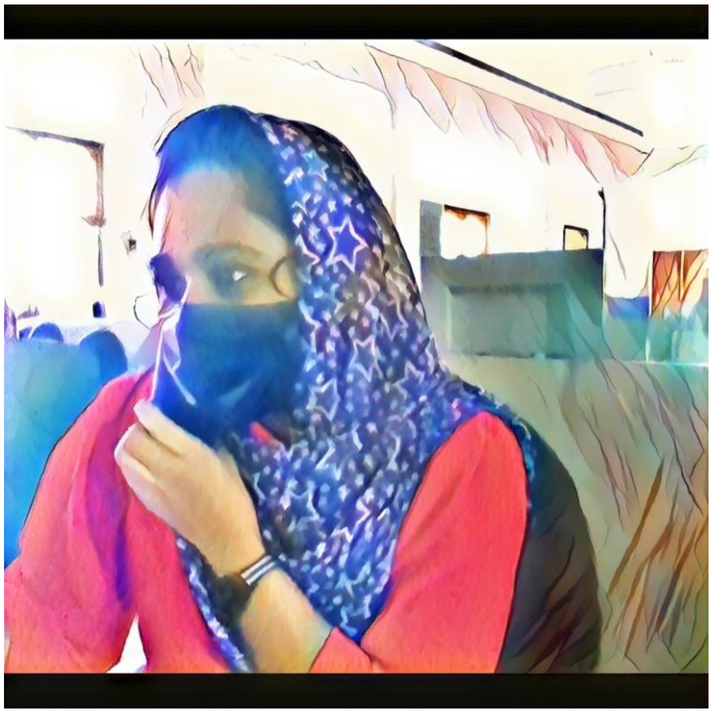


Figure4.6:Output of CartoonifyGAN

Chapter 5

Conclusion

It is proposed in the proposed study to use a CartoonifyGAN , Generative Adversarial Network to convert real-world pictures into cartoon-like visuals. A unique adversarial loss that favours clean edges is also proposed in addition to a sparse regularisation of high-level feature maps in the VGG-16 network for content loss, which enables smooth shading replication. The ultimate goal is to create images that are completely cartoon-like. Additionally, it suggests a simple setup process that is rapid to help with convergence. Tests have demonstrated that CartoonifyGAN is capable of learning a model that accurately and quickly converts photos of actual settings into cartoons, as illustrated in Figures 4.5, 4.6, and a decline in Figure 4.2's FID Score. It is significantly better than other "state-of-the-art" stylization methods.

5.1 Advantages

The main merits of proposed model are:

- Reduced FID score
- Compact architecture
- Fantastic cartoon-like images

5.2 Future Enhancement

- Experiments show that the CartoonifyGAN algorithm can produce amazing cartoon-style visuals..
- Shows an improvement in content loss (ie, FID score reduced is by1.14).
- Cartoonization's non-obviousness is a problem that needs to be handled in the future, and this work may potentially be extended to video animations.

References

[1] YongshengDong , Wei Tan, Dacheng Tao , LintaoZheng , and Xuelong Li , "**CartoonLossGAN: Learning Surface and Coloring of Images for Cartoonization**" in IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 31, 2022 485

[2]Y. Chen, Y.-K. Lai, and Y.-J. Liu, "**CartoonGAN: Generative adversarial networks for photo cartoonization,**" in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 9465–9474.

[3] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "**GANs trained by a two time-scale update rule converge to a local nash equilibrium,**" in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 6629–6640.

[4] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "**Unpaired image-to-image translation using cycle-consistent adversarial networks,**" in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 2242–2251.

[5] J. Chen, G. Liu, and X. Chen, "**AnimeGAN: A novel lightweight GAN for photo animation**," in Proc. Int. Symp. Intell.Comput. Appl., 2019, pp. 242–256.

[6] K. Simonyan and A. Zisserman. **Very deep convolutional networks for large-scale image recognition**.CoRR, abs/1409.1556, 2014.

[7] L. A. Gatys, A. S. Ecker, and M. Bethge, "**Image style transfer using convolutional neural networks**," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 2414–2423.

[8] F. Gao, X. Xu, J. Yu, M. Shang, X. Li, and D. Tao, **"Complementary, heterogeneous and adversarial networks for image-to-image translation**," IEEE Trans. Image Process., vol. 30, pp.3487–3498, 2021.

[9] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song, "**Neural style transfer: A review**," IEEE Trans. Vis. Comput. Graphics, vol. 26, no. 11, pp. 3365–3385, Nov. 2020.

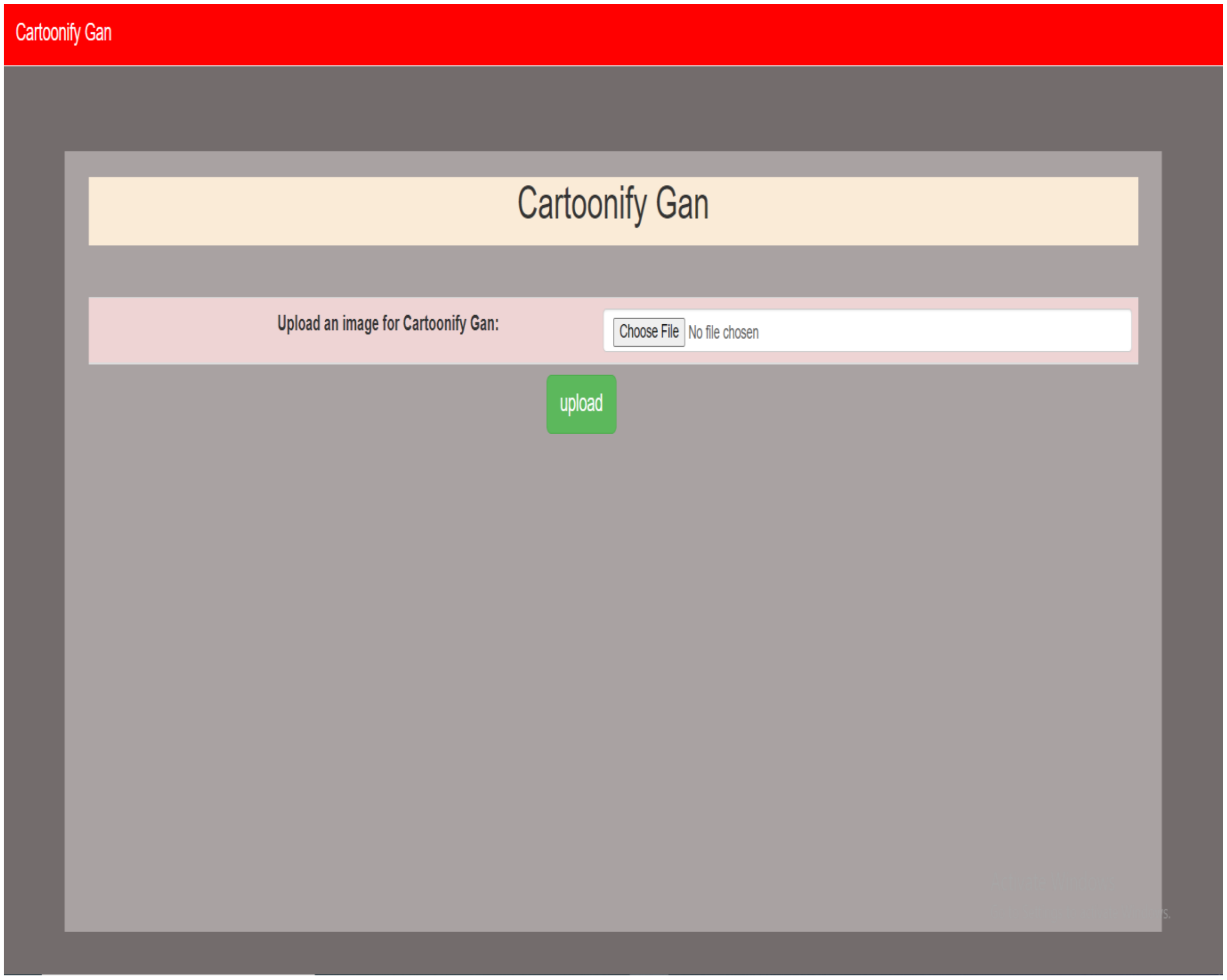
[10] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "**Image-to-image translation with conditional adversarial networks**," in Proc. IEEE Conf. Comput. Vis.Pattern Recognit. (CVPR), pp. 5967–5976, Jul. 2017.

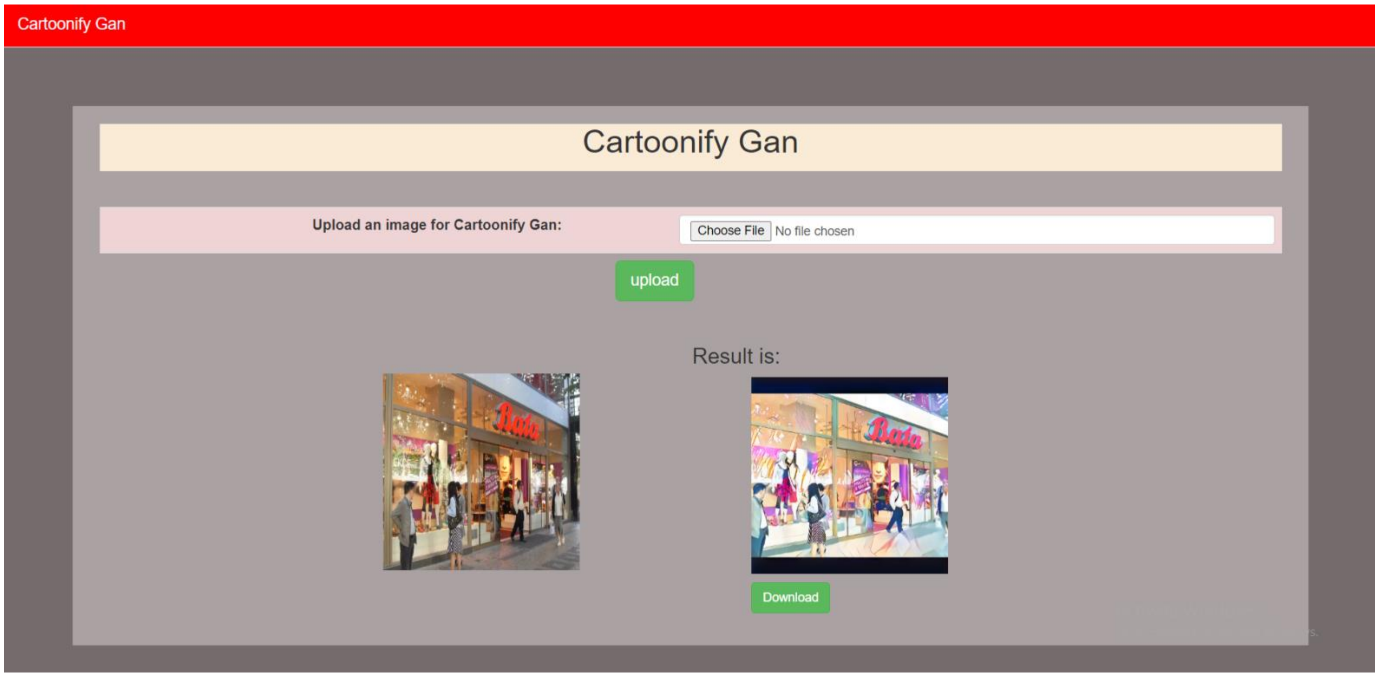
[11] Y. Jing et al., "**Stroke controllable fast style transfer with adaptive receptive fields**," in Proc. Eur. Conf. Comput. Vis., 2018, pp. 238–254.

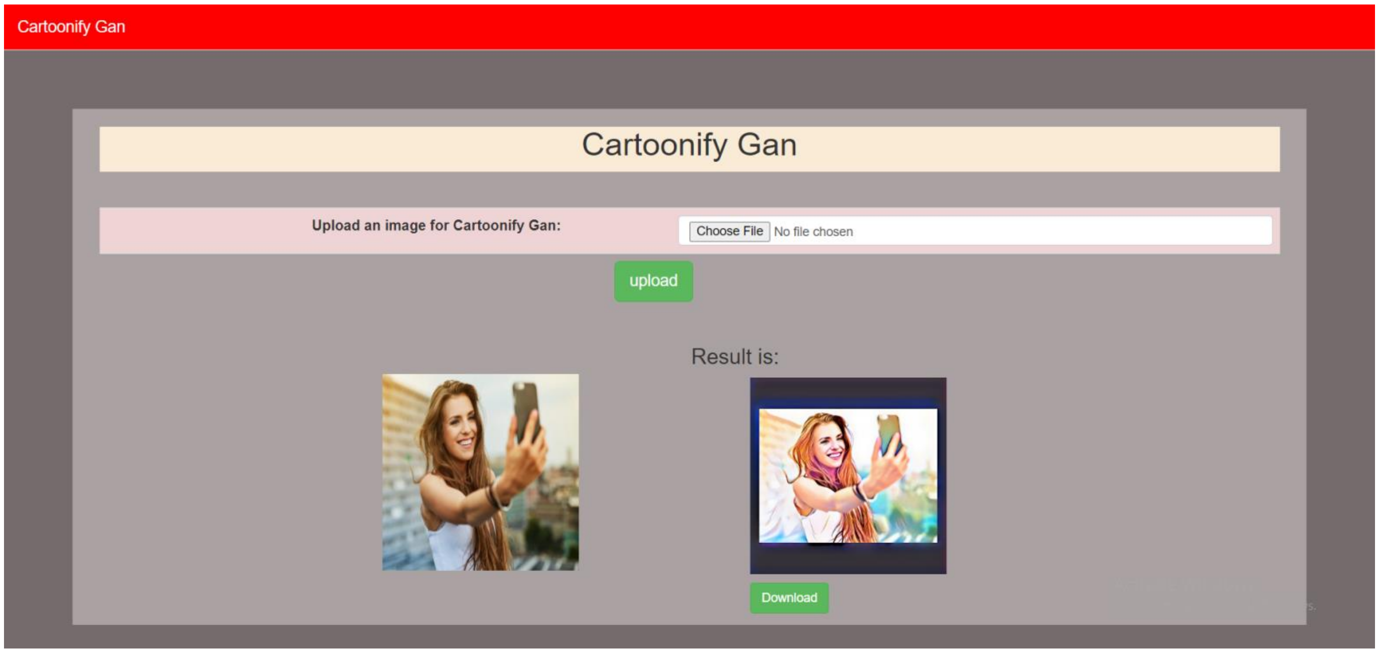
[12] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song, "**Neural style transfer: A review**," IEEE Trans. Vis. Comput. Graphics, vol. 26, no. 11, pp. 3365–3385, Nov. 2020.

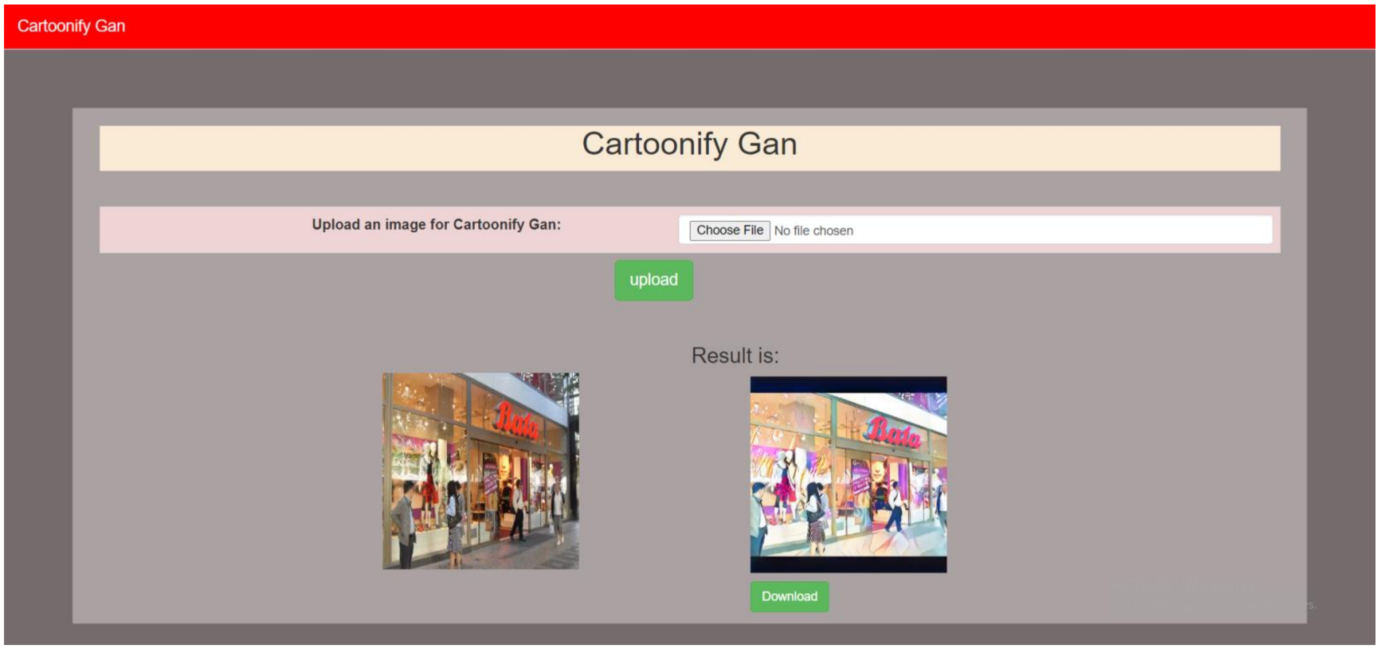
[13] I. Goodfellowet al., "**Generative adversarial nets**," in Proc. Adv. Neural Inf. Process. Syst., 2014, pp. 2672–2680.

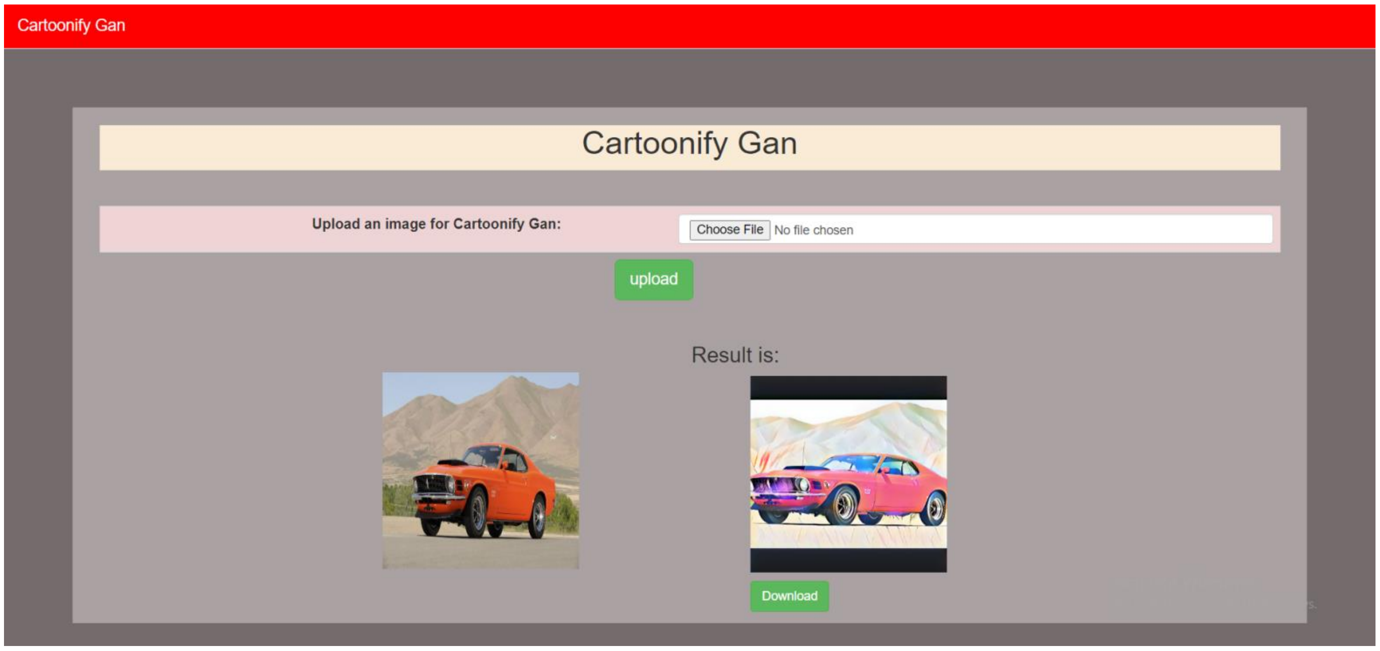
APPENDICES

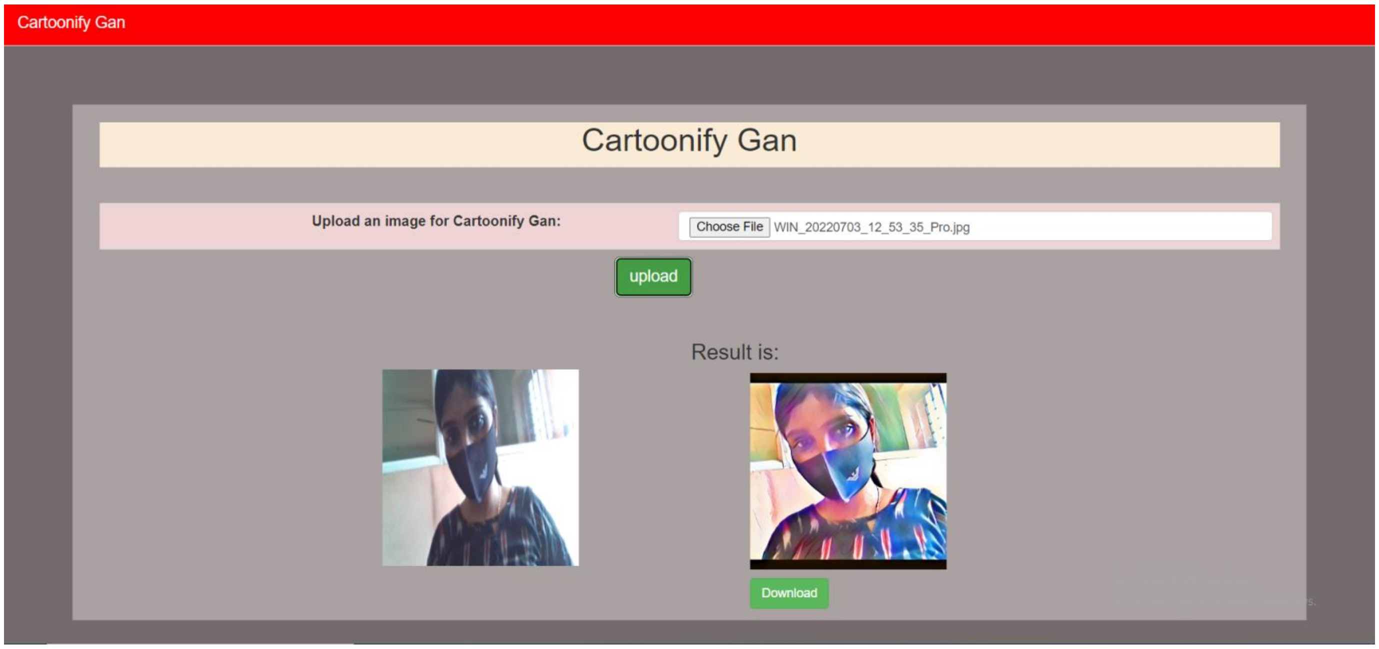

Cartoonify Gan
Cartoonify Gan
Upload an image for Cartoonify Gan:
Choose File
No file chosen
upload

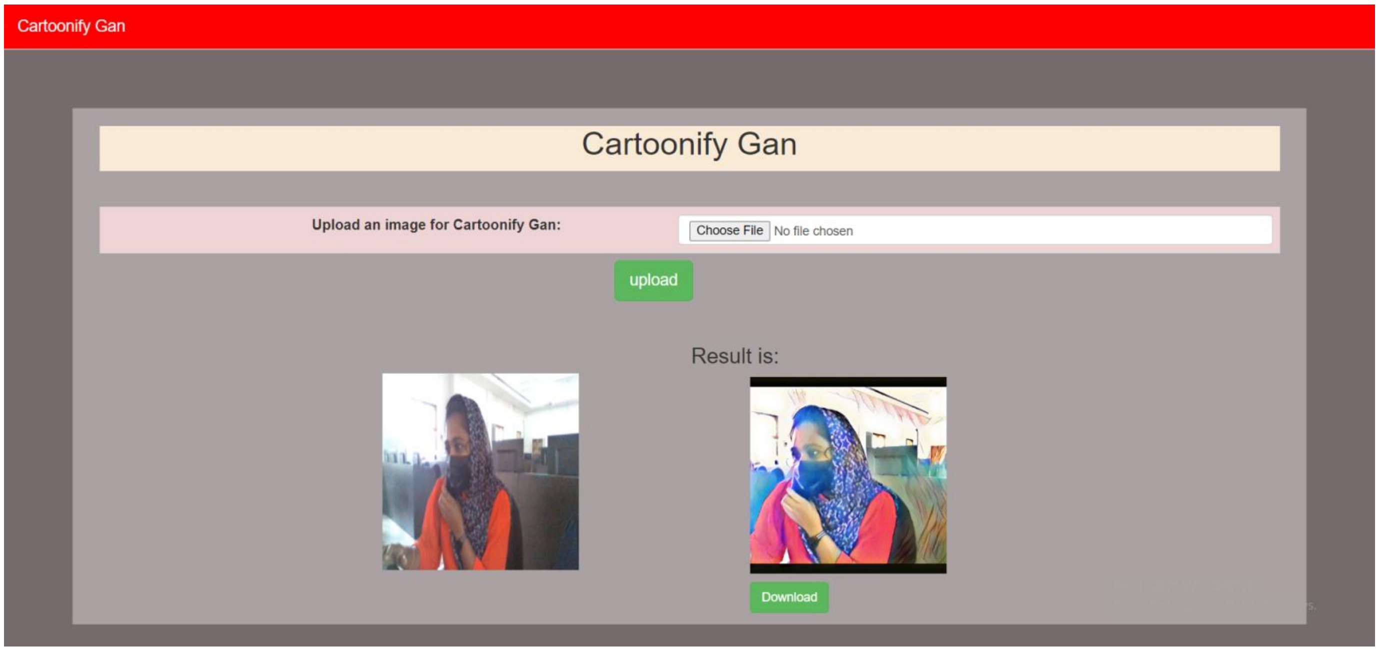

Cartoonify Gan
Cartoonify Gan
Upload an image for Cartoonify Gan:
Choose File
No file chosen
upload
Result is:
Download

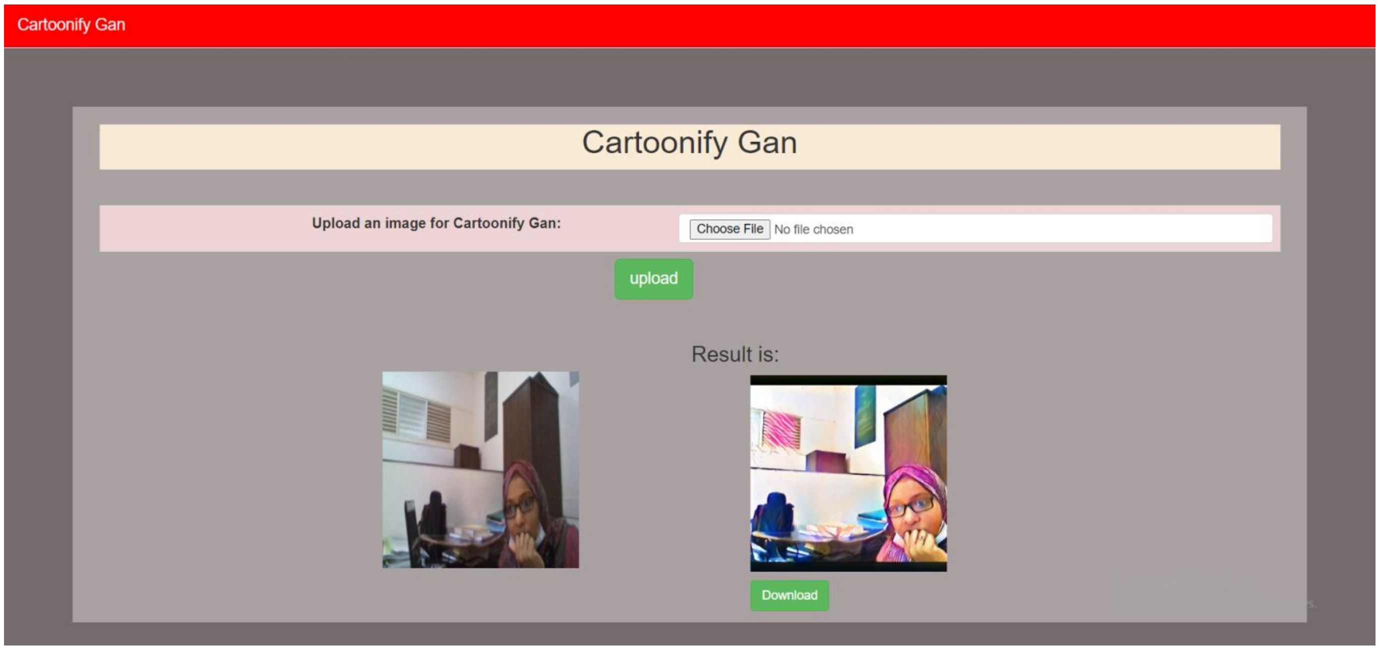

Cartoonify Gan
Cartoonify Gan
Upload an image for Cartoonify Gan:
Choose File
No file chosen
upload
Result is:
Download


Cartoonify Gan
Cartoonify Gan
Upload an image for Cartoonify Gan:
Choose File
No file chosen
upload
Result is:
Download


Cartoonify Gan
Cartoonify Gan
Upload an image for Cartoonify Gan:
Choose File
No file chosen
upload
Result is:
Download


Cartoonify Gan
Cartoonify Gan
Upload an image for Cartoonify Gan:
Choose File
WIN_20220703_12_53_35_Pro.jpg
upload
Result is:
Download


Cartoonify Gan
Cartoonify Gan
Upload an image for Cartoonify Gan:
Choose File
No file chosen
upload
Result is:
Download


Cartoonify Gan
Cartoonify Gan
Upload an image for Cartoonify Gan:
Choose File
No file chosen
upload
Result is:
Download