**Mini Project Report on**



**CARTOONIFY IMAGE USING MACHINE LEARNING**



**Submitted in partial fulfillment of the requirement for the award of the degree of**

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE & ENGINEERING**

**Submitted by:**

**Student Name : SHIVAM KUMAR**

**University Roll Number : 2019099**

***Under the Mentorship of***

**VIKASH TRIPATHI**

**Associate Professor**



**Department of Computer Science and Engineering**

**Graphic Era (Deemed to be University)**

**Dehradun, Uttarakhand**

**January 2023**



**CANDIDATE’S DECLARATION**

I hereby certify that the work which is being presented in the project report entitled **“Cartoonify image using machine learning”** in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineeringof the Graphic Era (Deemed to be University), Dehradun shall be carried out by the under the mentorship of **Vikash Tripathi, Associate Professor**, Department of Computer Science and Engineering, Graphic Era (Deemed to be University), Dehradun.

Shivam kumar 2019099 shivam

Name University Roll no. Signature

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Description** | **Page No.** |
| Chapter 1 | Introduction | 4 |
| Chapter 2 | Literature Survey | 5-6 |
| Chapter 3 | Methodology | 7-8 |
| Chapter 4 | Result and Discussion | 9-10 |
| Chapter 5 | Conclusion and Future Work | 11-12 |
|  | References |  |

**Chapter 1**

**Introduction**

This report presents the method that is used to convert an input image to the desired

cartoon image. It is a very interesting project which is simple and basic, made for

learning purpose. we can modify this project as per our requirements and develop

a perfect advanced level application in future.

* 1. **Problem Statement**

The process of creating a cartoon-like effect is time-consuming and requires a lot of space. Currently, there are no simple solutions to provide cartoon-like effects to images. In some cases, users need to install complex photo editing software, such as Photoshop, while in other cases, they have to perform some tasks themselves.

There are many existing website to perform such tasks, however, it is difficult to use as users have to mark down points and lines on the image to apply effects, which isn’t user-friendly and the options are limited. A user-friendly website that performs the task of applying effects to images very well is therefore required.

With this Machine Learning Project, we will be building a system that will convert any image into its cartoon version. We are going to use OpenCV for this project. This project will take an input image and the result will give us a cartoonified version of the image. This project sounds fun. Any artist would take hours and a lot of money for doing this, but with this project, we can do this for free. We can also convert this project into an app and made this

available for users.

**Chapter 2**

**Literature Survey**

**2.1**

**Title :** A Neural Algorithm of Artistic Style (Leon A. Gatys, Alexander S. Ecker, Matthias Bethge)

**Published Year:** 2015

**Algorithm Used :** An artificial system based on a Deep Neural Network that creates artistic images of high perceptual quality. we take pre-trained neural network and define a 3 Component loss function that will enable us to achieve our end goal of style transfer and then optimize over that loss function.

**Benefit :** First Ever Successful Attempt to create Artistic style transfer using Deep Neural Network. All previous attempts/algorithms to achieve style transfer had failed. The system uses neural representations to separate and recombine content and style of arbitrary images, providing a neural algorithm for the creation of artistic images.

**Constraints :** While the Gatys et al. method produced beautiful neural style transfer results, the problem was that it was quite slow.

**2.2**

**Title :**  Perceptual Losses for Real-Time Style Transfer and SuperResolution (Justin Johnson, Alexandre Alahi, Fei-Fei Li)

**Published Year :** 2016

**Algorithm Used :** proposes the use of perceptual loss functions for training feed-forward networks for image transformation tasks. a feed-forward network is trained to solve the optimization problem proposed by Gatys et al. in real-time.

**Benefit :** Johnson et al. (2016) built on the work of Gatys et al., proposing a neural style transfer algorithm that is up to three orders of magnitude faster. The Johnson et al. method frames neural style transfer as a superresolution-like problem based on perceptual loss functions.

**Constraints :** While the Johnson et al. method is certainly fast, the biggest downside is that you cannot arbitrarily select your style images like you could in the Gatys et al. method.

**2.3**

**Title :** Instance Normalization: The Missing Ingredient for Fast Stylization (Dmitry Ulyanov, Andrea Vedaldi, Victor Lempitsky)

**Published Year:** 2017

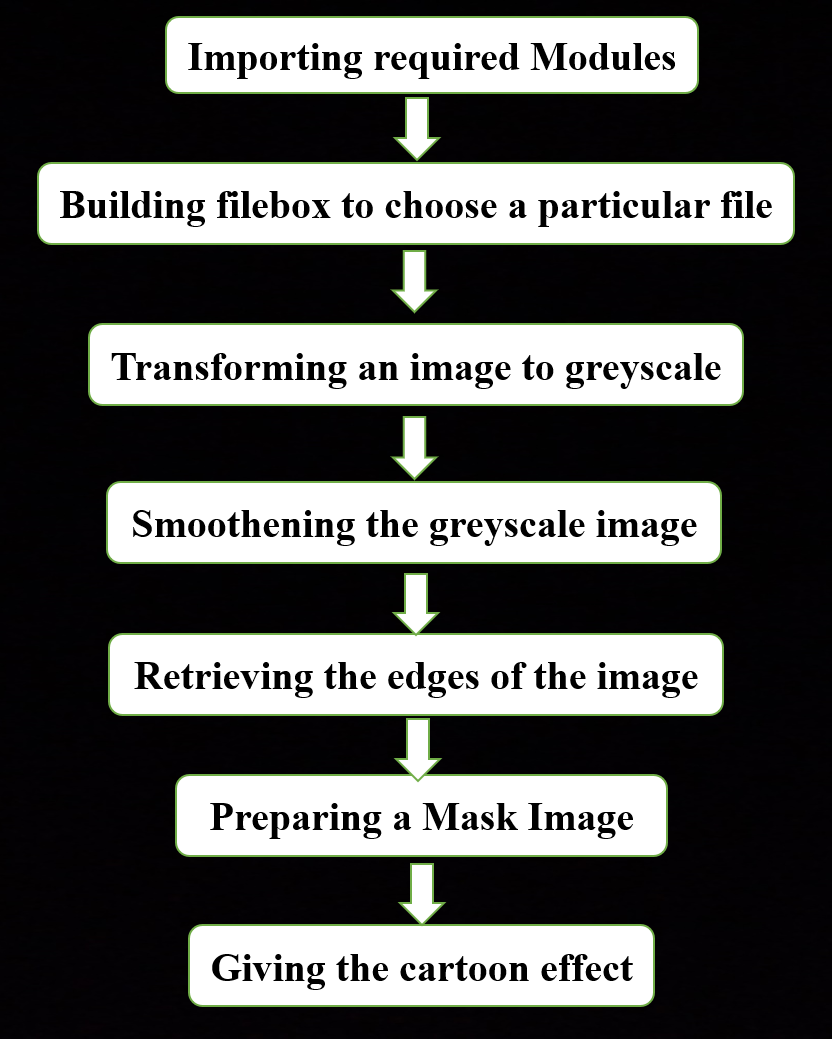
**Algorithm Used :** The change is limited to swapping batch normalization with instance normalization, and to apply the latter both at training and testing times. The resulting method can be used to train high-performance architectures for real-time image generation.

**Benefit :** it was found that swapping batch normalization for instance normalization (and applying instance normalization at both training and testing) in Gatys et al. method, leads to even faster real-time performance and arguably more aesthetically pleasing results as well.

**Constraints :** None

**Chapter 3**

**Methodology**



Above methodology we are going to follow to convert our input image to required cartoonified image.

1. At first we will import required python libraries in our source file, then we will build the filebox such that we can choose a particular file , then the image is converted to a greyscale image.
2. The next two steps are the important steps to converting images into cartoon images. They are smoothening the greyscale image and then retrieving the edges.
3. In this color of the image is smoothened to give the cartoon look and then we retrieve the edges and then highlight them in the final image.
4. Next, we will prepare a mask Image. In this, we use the bilateral filter with removes the noise and smoothen it to some extent.
5. Now the final step is giving the cartoon effect. To the image which we got in the previous step, we combine our two important steps and finally give a mask-edged image that looks like a cartoon image.

**Chapter 4**

**Result and Discussion**

**INPUT :**



**OUTPUT** 

**Discussion**

By applying the algorithm to cartoonify the image using machine learning with python as our programming language , we have successfully created our required cartoon image.

**Chapter 5**

**Conclusion and Future Work**

In this Machine Learning project, we learned how to cartoonify an image. In this project, we have used OpenCV and we used the black and white edge of our original image and used it as a mask on our original image.

Thus we have shown that how image can be converted to cartoon. We also stated the examples on how image is converted to cartoon. Python libraries requirements for image to cartoon conversion are also shown in this paper. The systematic working of image to cartoon conversion and respective algorithm is shown with a flowchart in this paper. Also we have stated challenges and problems one can face while cartoonifying the captured image. In this paper we have also discussed need and scope of cartoonifying the content image

**Future Work**

The goal of this project is to create an Android app with a simple user interface allowing users to apply the cartoon algorithm to images of their choice. The algorithm is designed to provide artistically and comically appealing results on as wide a range of pictures as possible, although it is conceded that not all inputs will yield equally satisfying results.

**References**

[1] N. K. Kanhere and S. T. Birchfied, “Real-time incremental segmentation and tracking of vehicles at low camera angles using stable features,” *IEEE Trans. Intell. Transp. Syst*., vol. 9, no. 1, pp.148-160, March 2008 **(Example : Journal papers)**

[2] K. Onoguchi, “Moving object detection using a cross correlation between a short accumulated histogram and a long accumulated histogram”, Proc. 18th Int. Conf. on Pattern Recognition, Hong Kong, August 20 - 24, 2006, vol. 4, pp. 896 – 899 **(Example : Conference papers)**

[3] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, “Introduction to Algorithms”, 2nd ed., The MIT Press, McGraw-Hill Book Company, 2001 **(Example : Text Book/ Magazine)**

[4]Open Source Computer Vision (OpanCV) [Online]. Accessed on 21st April 2022: <http://opencv.willowgarage.com/wiki/> **(Example : Website)**