RESEARCH ARTICLE OPEN ACCESS

PICTURE AND MPEG CARTOONIZATION UTILIZING OPEN EXCEL AND AI

SEELAM ABHILASH REDDY,UG scholar,CSIT,Sri Indu College Of Engineering & Technology(A)
D RITHISHA,UG scholar,CSIT,Sri Indu College Of Engineering & Technology(A)
PILLALAMARRI LAXMI PRAGNA VARUN,UG scholar,CSIT,Sri Indu College Of Engineering & Technology(A)
JAKKULA RAMESH,UG scholar,CSIT,Sri Indu College Of Engineering & Technology(A)
Mrs M. Sudarani,Asst.Prof.CSIT,Sri Indu College Of Engineering & Technology(A)

ABSTRACT

Cartoonization of images and videos could be used in various different applications, which can be ease in publishing a comic book for a comic, anime, T.V. shows as well as for fun events on social media. This paper proposes cartoonization of images and videos through Generative Adversarial Networks (GANs). Thus an idea to convert realworld images and videos into cartoonized one is proposed through this paper. With carbonization, the paper also proposes to make a complete Image-hub for the user with features including upscaling, denoising and editing filters to the input images through the Python OpenCV library. The project also includes video to GIF conversion touse in various social media platforms to achieve cartoon filters. Thus the project is built to be user friendly and leveraging various other features rather than only cartoonization of images and videos.

I.INTRODUCTION

Cartoon is an image or series of images that are formed using a sequence of illustrations for animations. These cartoons may represent realistic or non-realistic features. However cartoons have gained a huge attention especially by the children, teenagers and artists. Due to which there exists many applications where cartoons are used. Some of these applications include cartoon television shows, comic magazines, cartoon based image filters and animated films. Some of the applications may also contain some real-world scenes. For example, an animated film may contain an image having a city drawn which corresponds to a city that is present in a real world. These cartoon images are created by a skillful artist manually drawing those scenes or by using computer software's to create a single image. To obtain a better quality, the artists need to draw lines and must shade each color region based on real-life scenes. This entire process requires a lot of labor skills and is really time consuming especially while working on animated comics or films. Also the existing computer software's like Corel Draw or Adobe Photoshop are not free to use and also may not be easy for the beginners to understand and achieve the required quality. So there is a requirement of technology that can help transform a real-world based photo or video into an animated image or video respectively. This technology when integrated with other software's can help the user to convert their real-world photos or videos into cartoon versions as and when required or can also act as an image filter which is also freely available and easy to use. In this paper, we propose a Generative Adversarial Networks (GANs) based approach along with features like image denoising and image upscaling to convert an image, GIF or video files into their cartoon versions. The image upscaling and denoising is achieved using OpenCV. To train the model, data used are a set of photos and a set of cartoon images. The trained model helps in generating the cartoon images or videos that are not a part of training data.

II.REVIEW OF LITERATURE

The huge achievement has been accomplished with learning-based stylization, cutting edge techniques neglect to deliver Cartoonized pictures with satisfactory quality. There are two reasons. To begin with, rather than adding surfaces, for example, brush strokes in numerous different styles, animation pictures are exceptionally streamlined and disconnected from genuine world photographs. Second, in spite of the variety of styles among specialists, animation pictures have a recognizable regular appearance — clear edges, smooth shading concealing, and general basic surfaces which are altogether different from different types of works of art [2]. Customary strategies are essentially founded on misrepresented numerical theoretical models. These techniques regularly apply face parsing strategies to section out every facial segment; at that point utilize non-photorealistic delivering strategy or basic sifting handling to get animation pictures. In view of these strategies, there are different picture cartoonizations APPs on our cell phone, like MomanCamera, Cartoon Camera Photo

very time-consuming. An effective computer program to transform photos of real world scenes to comic styles will be a very useful tool for artists to build their work on. In addition, such techniques can also be integrated into photo editing software such as Photoshop and Instagram, for turning everyday snapshots into comic styles [1]. To improve the results, alternative methods rely on segmentation of images/videos , although at the cost of requiring some user interaction. Dedicated methods have also been developed for portraits, where semantic segmentation can be derived automatically by detecting facial components. However, such methods cannot cope with general images[1][4].

INTRODUCTION TO GAN

Generative Adversarial Networks (GANs) are a powerful class of neural networks that are used for unsupervised learning and machine learning models capable of generating realistic image, video, and voice outputs. Established in-game hypothesis, GANs have far-reaching applications: from improving online protection by battling against adversarial assaults and anonymizing data to safeguard security to producing best in class pictures, colorizing high contrast pictures, expanding picture goal, making symbols, turning 2D pictures to 3D, and that's just the beginning. A generative adversarial network (GAN) has two parts:

- The generator learns to generate plausible data. The generated instances become negative training examples for the discriminator.
- The discriminator learns to distinguish the generator's fake data from real data. The discriminator penalizes the generator for producing implausible results.

When training begins, the generator produces obviously fake data, and the discriminator quickly learns to tell that it's fake [9].

GAN Architecture

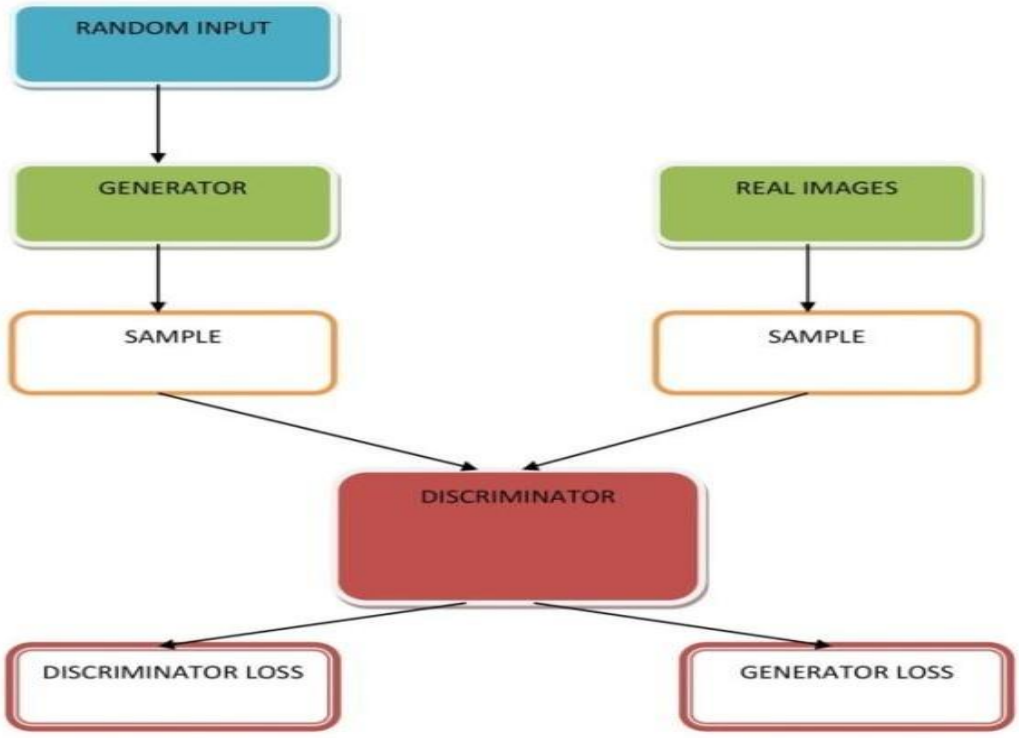

Fig. 1: Block Representation of GAN Architecture

Discriminator

The discriminator in a GAN is simply a classifier. It tries to distinguish real data from the data created by the generator. It could use any network architecture appropriate to the type of data it's classifying. The discriminator model takes an xample from the domain as input and predicts a binary class label of real or fake (generated).The real example comes from the training dataset. The generated examples are output by the generator model. The discriminator is a normal classification

model. After the training process, the discriminator model is discarded as we are interested in the generator [10]. During discriminator training: 1. The discriminator classifies both real data and fake data from the generator. 2. The discriminator loss penalizes the discriminator for misclassifying a real instance as fake or a fake instance as real. 3. The discriminator updates its weights through back propagation from the discriminator loss through the discriminator network.

Generator

The Generator is trained while the Discriminator is inactive. After the Discriminator is trained by the produced fake data of the Generator, we can get its predictions and use the results for training the Generator and improve from the past state to attempt to trick the Discriminator. Generator training requires tighter integration between the generator and the discriminator
than discriminator training requires [10]. The portion of the GAN that trains the generator includes:

- Random input.
- Generator network, which transforms the random input into a data instance.
- Discriminator network, which classifies the generated data.
- Discriminator output.
- Generator loss, which penalizes the generator for failing to fool the discriminator.

III.MAINTAINING QUALITY OF IMAGES USING OPENCV LIBRARY

OpenCV (Open Source Computer Vision Library) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel). The library is cross-platform and free for use under the open-source Apache 2 License. OpenCV features GPU acceleration for real-time operations [7].

Image Denoising using OPENCV

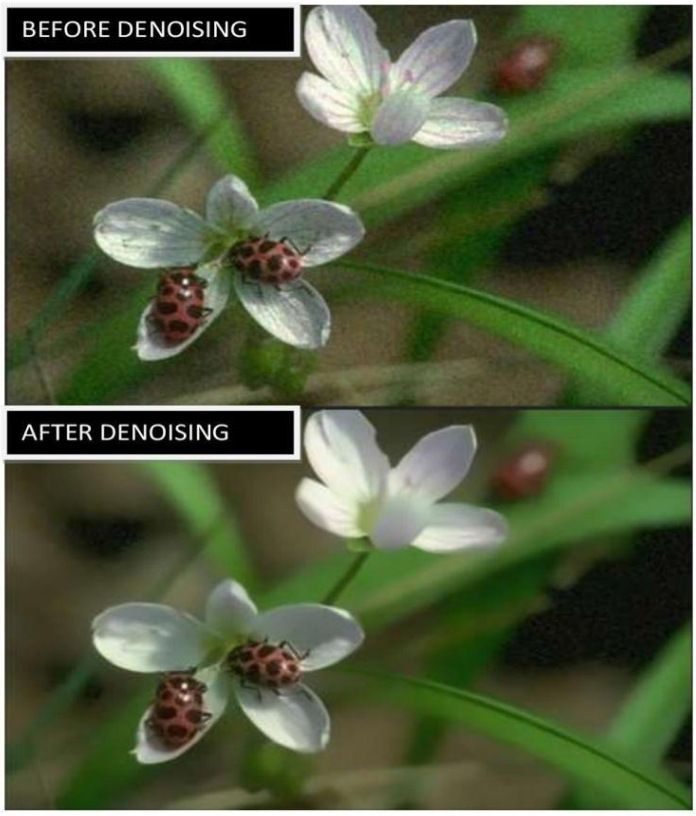

Fig. 2: Results for image denoising

One of the basic challenges within the field of image process and computer vision is removal of unnecessary things in an image file which is generally termed as image noise. Image noise could also be caused by totally different intrinsic (i.e., sensor) and accidental (i.e., environment) conditions. To remove those unnecessary things in an image, image denoising plays a vital role in a very wide selection of applications like image restoration, visual pursuit, image registration, image

segmentation, and image classification. The underlying goal is to estimate the original image by suppressing noise from a noise-contaminated version of the Image. To obtain this image denoising feature, the OpenCV library contains a function fast NlMeans Denoising Colored which input converts image to CIELAB colorspace and then separately denoise L and AB components with given parameter regulating filter strength for luminance component [8].

Image Upscaling using OPENCV

When scaling a vector graphic image, the graphic primitives that conjure the image can be scaled victimization geometric transformations, with no loss of image quality. When scaling a formation graphics image, a different image with an improved or lower vary of pixels must be generated. at intervals, the case of decreasing the image part varies (scaling down) this usually finally ends up during a plain quality loss. The two techniques which we applied are EDSR (Enhance Deep Super-Residual Network) and FSRCNN (Fast Super Residual Convolutional Neural Network). EDSR takes up to 120 seconds to upscale the image whereas FSRCNN is a faster technique and gives an upscaled image output within 10 seconds. But comparison wise EDSR gives a higher upscaled image than FSRCNN [5].

Fig. 3: Results for image upscaling

IMPLEMENTATION OF THE PROJECT

CartoonGAN can deliver great cartoon stylization utilizing the information of individual artists for preparing, which are effortlessly acquired from the cartoon videos, since our technique doesn't need matched pictures. Here are the methods we have proposed in our system.

Data

The training data contains real-world original photos and cartoon images, and the test data only includes real-world photos. All the training images are resized and cropped to 256×256. Photos. 6,153 photos are downloaded from Flickr, in which

Cartoonization of Images using GAN

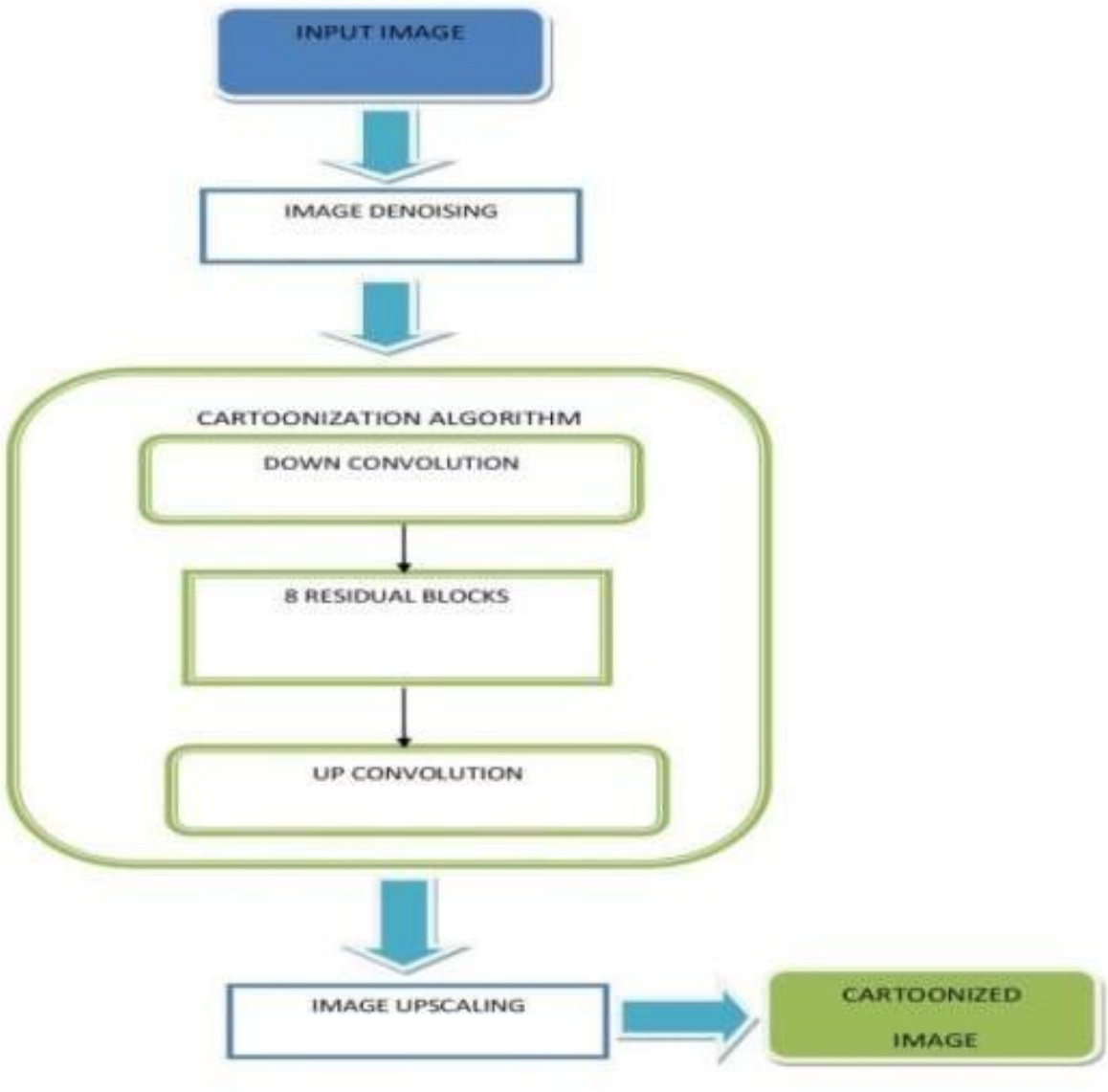

Fig. 4: Block Representation Proposed Image Cartoonization Method

The Image will be first Denoised and then it will follow the cartoonization Algorithm. In Cartoonizing Images, the generator network is utilized to map input pictures to the animation complex. Cartoon stylization is created once the model is prepared. The generator starts with a flat level convolution stage followed by two down-convolution squares to spatially pack and encode the pictures. Valuable local signals are separated in this stage for downstream change. Afterward, eight remaining squares with indistinguishable formats are utilized to build the substance and complex element [2]. At last, the output cartoon-style pictures are reproduced by two up-convolution blocks and it will be up scaled in order to increase the quality of image.

A. Converting cartoonized video to GIF file

Gif Conversion method will follow the proposed cartoonization of video algorithm where only Cartoonization of video will take place and Sound Extraction will be discarded. The Input Gif will be first converted into a video file. The Video File will be cartoonized by Cartoonization Algorithm. At last the Video will be converted into GIF file. The Cartoonized video can easily be converted into GIF by applying a simple code which uses python imageio library.

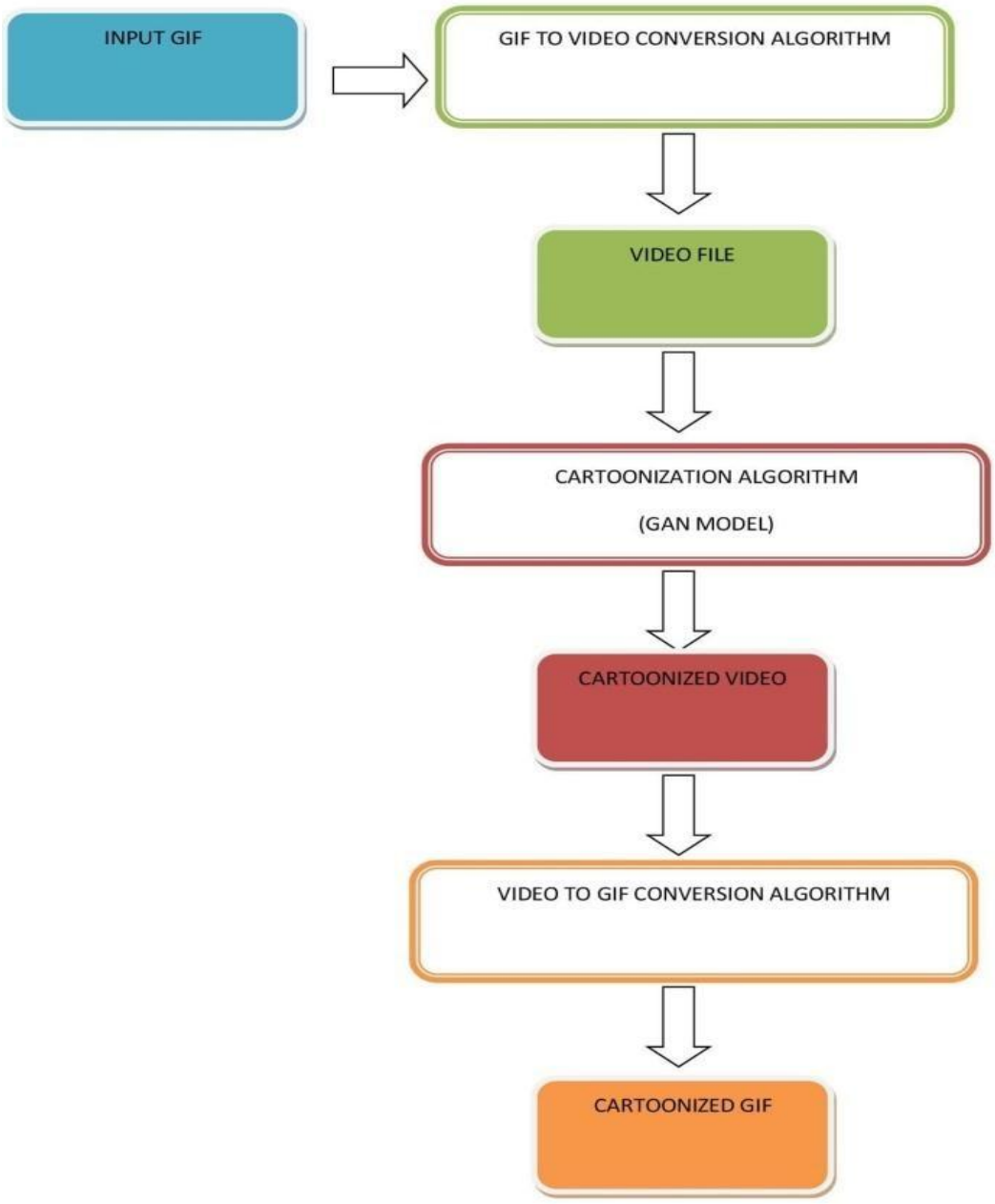

Fig. 5: Block Representation of Proposed GIF Cartoonization Method

Cartoonization of videos using GAN

Whenever we give an input video for cartoonization, the video gets converted into frames and each frame is Cartoonized individually. The process of cartoonization for frames is same as that of images. Since the video gets converted into frames, loss of sound is observed. In order to aid this problem we have come up with a method so that there will be no audio loss. Here we apply the concept of multithreading. The input video will send its copy to the sound loss prevention and control unit so that the sound will be extracted from the video and an audio file will be created. At the same time the process of dividing video into frames will take place. Each frame will be Cartoonized. After the cartoonization of all the frames, the frames will be combined to form the Cartoonized video. At last both the outputs from Sound Loss Prevention Control Unit and Video Cartoonizing unit will go through Sound Merging Algorithm where the audio and video will be merged. The output of this whole process would result in a Cartoonized video which will have same audio as that of the input video.

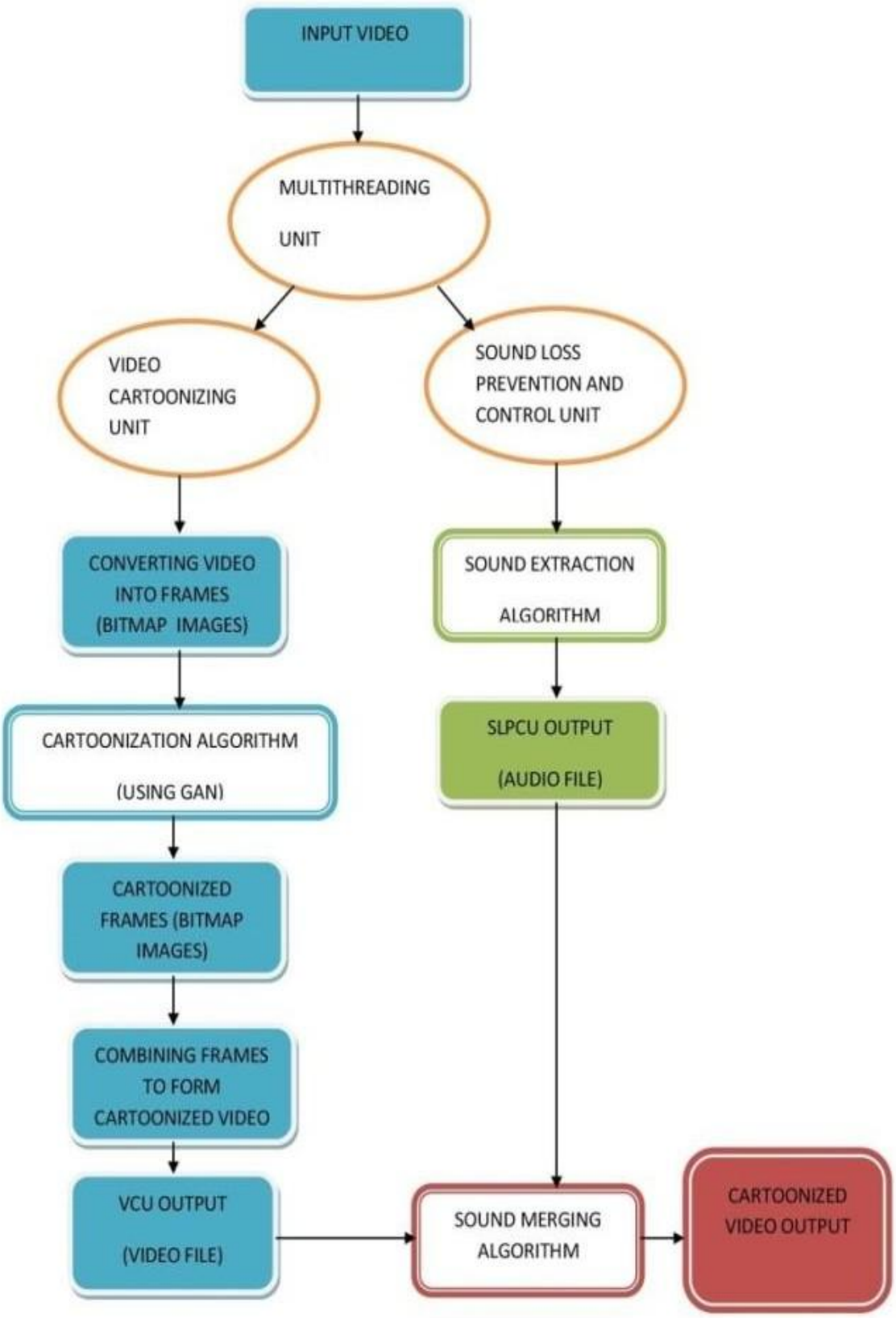

Fig. 6: Block Representation of Proposed Video Cartoonization Method

Results

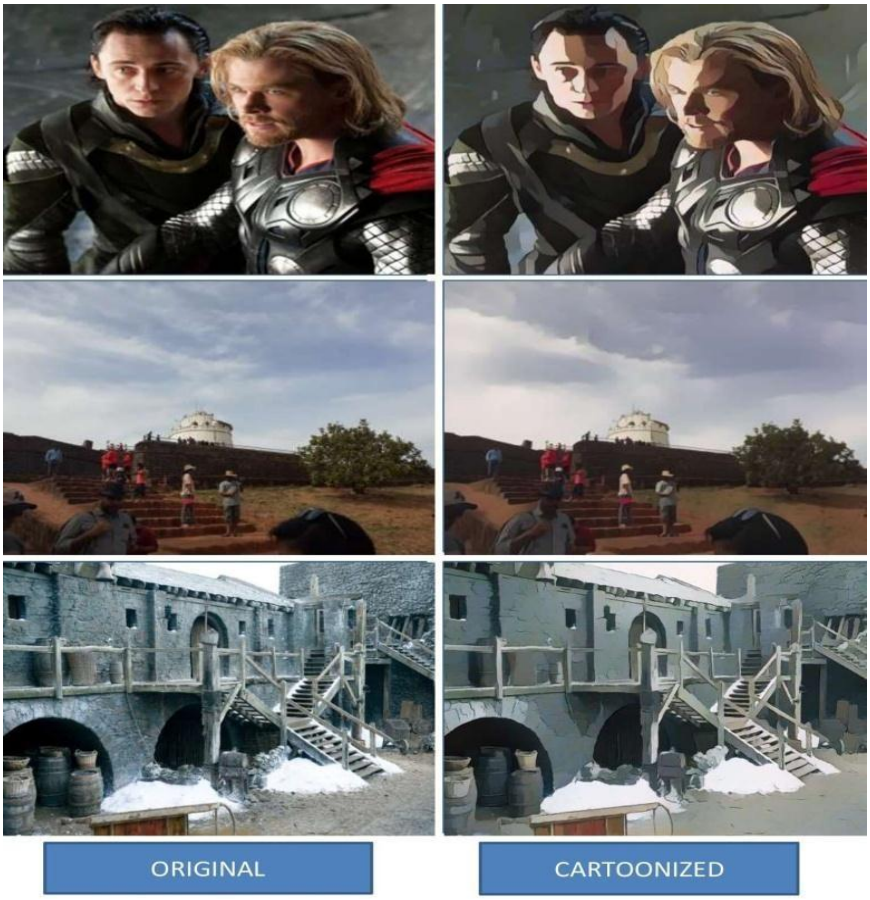

Fig. 7: Results for Image Cartoonization

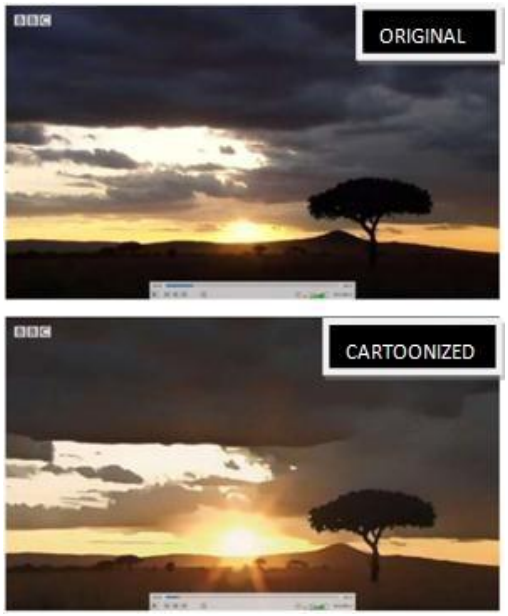

Fig. 8(a): Result-1 for Video Cartoonization

Fig. 8(b): Result-2 for Video Cartoonization

IV.CONCLUSION

The present methods for converting real-world images or videos into cartoon versions which were proposed by other systems compromise the quality of the image. In the case of video, the audio present in the video file is lost in the resultant cartoon version of the video. We purpose a system that helps in cartoonization of images and videos with the help of Generative Adversarial Models (GANs). To implement this, real-world image files are denoised and then passed through the GAN model which generates the desired cartoonized image. The video is cartoonized by dividing the video into multiple image frames and simultaneously extracting audio from the image file. Each image frame is denoised and passed through the GAN to generate frames of cartoonized image. Those cartoonized image frames are aggregated and converted into video which is aggregated with the audio file to obtain the cartoonized videos. After the implementation of the project, the desired cartoon version of images and videos was achieved. Also the loss of audio has been solved using this approach. Hence, the GAN based Cartoonization model helps in saving time to convert the real-time image/videos into their cartoon version with less noise and better quality

V.REFERENCES

[1] Y. Chen, Y.-K. Lai, Y.-J. Liu, "Transforming photos to comics using convolutional neural networks", InternationalConference on Image Processing, 2017.

[2] Yang Chen, Yu-Kun Lai, Yong-Jin Liu, “CartoonGAN: Generative Adversarial Networks for PhotoCartoonization”,2020.

[3] I..Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. WardeFarley, S. Ozair, A. Courville, and Y. Bengio,“Generative adversarial nets,” in Advances in neural information processing systems, 2014, pp. 2672– 2680.

[4] Akanksha Apte, Ashwathy Unnikrishnan, NavjeevanBomble, Prof. SachinGavhane,” Transformation of RealisticImages and Videos into Cartoon Images ”,2020

[5] Bee Lim, Sanghyun Son, Heewon Kim Seungjun, Nah Kyoung Mu Lee,”Enhanced Deep Residual Networks forSingle Image Super-Resolution”, arXiv:1707.02921v1 [cs.CV], 10 Jul 2017

[6] Chao Dong, Chen Change Loy, Xiaoou Tang, “Accelerating the Super-Resolution Convolutional Neural Network”, arXiv:1608.00367v1 [cs.CV], 1 Aug 2016

[7] https://opencv-pythontutroals.readthedocs.io/en/latest/py_tutorials/py_setup/py_intro/py_intro.html

[8] https://docs.opencv.org/3.4/d1/d79/group__photo_denoise.html#ga03aa4189fc3e31dafd638d90de335617

[9] https://www.geeksforgeeks.org/generative-adversarial-network-gan/

[10] https://developers.google.com/machine-learning/gan/gan_structure