IMAGE TO SKETCH CONVERTER

SUBMITTED BY

SANDHYA M (711721243092)

RUJUTA M (711721243083)

RAGHUL VISHAL T (711721243074)

VAISHNAVI R (711721243119)

Under the guidance of

Mrs. G. ELIZABETH RANI (M. Tech)
(Assistant Professor, Department of Computer Science and Business Systems)

In partial fulfilment for the award of the degree Of

BACHELOR OF TECHNOLOGY

In

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

KGISL INSTITUTE OF TECHNOLOGY
(Affiliated to Anna University)
Saravanampatti, Coimbatore – 641035.

Academic Year 2022-2023

KGISL INSTITUTE OF TECHNOLOGY

(AFFILIATAED TO ANNA UNIVERSITY)
Saravanampatti, Coimbatore – 641035

BONAFIDE CERTIFICATE

This is to certify that the Mini project titled "**IMAGE TO SKETCH CONVERTER**" is a Bonafide record of the work done by SANDHYA M (21AIB29), RUJUTA M (21AIB21), RAGHUL VISHAL T(21AIB11), VAISHNAVI R(21AIB55), in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Artificial Intelligence And Data Science , during the academic year 2022-2023.

Mrs. G. ELIZABETHRANI — Dr. KALPANA S

Supervisor — HOD

Assistant Professor/ CSBS — Department of AI&DS

Submitted for the University Viva-voce Examination held on 24.05.2023.

ACKNOWLEDGEMENT

First and foremost, I wish to thank god for his grace to complete this "**IMAGE TO SKETCH CONVERTER**" project successfully. I would also thank my special thanks to my dear parents from the bottom of my heart for their support in making our project complete.

I express a deep sense of gratitude to **Dr. Ashok Bakthavathsalam,** Founder chairman, and a special thanks to **Dr. Shankar** principal, and **Dr. Suresh Kumar** Vice Principal, and also a special thanks to **Ms.Kalpana S** Professor& Head of the Department of Artificial Intelligence And Data Science, KGISL Institute for granting the permission and providing necessary facilities to carry out project work.

I am highly indebted to KGISL Institute of Technology (KITE) and express my special and sincere thanks to my passion project supervisor **Mrs. G. ELIZABETH RANI** Assistant Professor, CSBS of KGISL Institute of Technology for her inspiring guidance, constant encouragement with my work during all stages. I am feeling very glad to do the project under her guidance, which truly practices and appreciates deep thinking. I will be forever indebted to my guide for supporting me in difficult times, when started this project and during compilation errors, she gave moral support and let this project move on.

ABSTRACT

The project Image-to-Sketch Converter seeks to create a cutting-edge artificial intelligence (AI) system that can convert digital photos into creative sketches. In order to automate the process of creating sketch-like representations from photographic inputs, this project leverages the capabilities of deep learning techniques and computer vision algorithms.

The suggested method makes use of a convolutional neural network (CNN) architecture that has been trained on a sizable dataset of matched pictures made up of authentic photographs and the accompanying hand-drawn sketches. This develops high-level features and distinctive patterns that encapsulate the essence of a sketch through an iterative training process, enabling it to produce precise and aesthetically pleasing sketch outputs.

Numerous uses for the image-to-sketch converter are available, such as artistic expression, image editing, and design. With the help of this application, artists may quickly create representations of their digital images that resemble sketches, giving their visual compositions a new perspective. This approach can also be used by graphic designers to produce original sketches for logos, graphics, and other design components, improving the aesthetic appeal overall.

Both objective and arbitrary criteria are used to assess the image-to-sketch converter. Comparing generated sketches to actual sketches allows for the measurement of correctness and fidelity using objective metrics, whereas user tests and surveys used for subjective evaluation entail human perception and judgement. Results from experiments show that the suggested approach performs very well, creating high-quality sketches with accurate replication of details and maintaining the creative character of the input.

CHAPTER 1

INTRODUCTION

Welcome to the creative and artistic world! We are pleased to introduce our ground-breaking project, an Image to Sketch Converter, to you in this modern age where the distinction between fact and fantasy is becoming hazy. Our initiative intends to close the gap between digital and traditional art forms by utilising the strength of cutting-edge algorithms and artificial intelligence. With the aid of this ground-breaking tool, you can turn any photograph into a captivating hand-drawn sketch that perfectly captures the spirit and feeling of the original image. Our Image to Sketch Converter is the entrance to a mesmerising visual domain, whether you're an artist looking for fresh inspiration or a photography enthusiast wanting to add a little artistic flair to your photos. As we unleash the amazing potential, join us on this extraordinary journey.

This technique makes it easier for people to comprehend how painters create pencil sketches, which are one of the most basic pictorial languages for representing the abstract human sense of natural scenes. In addition, presenting realistic scenes in a non-lifelike, creative approach is frequently desired in the gaming and film industries because creators have discovered that by making pictures appear less photorealistic, audiences can feel more immersed in a narrative.

As a result, computers assist in reducing the amount of labour that would otherwise be required to produce a non-realistic video segment by having artists manually sketch several images. Last but not least, these algorithms enable regular individuals to "become artists". People's lives would be made much more enjoyable if they could turn common images they take themselves into original works of art, making this technology commercially viable.

The methods from the paper Combining Sketch and Tone for Pencil Drawing Production were actually attempted to be re-implemented in this project. Edge detector serves as the starting point for sketching object contours during the stroke-drawing step. Then, to both categorise and depict the edges, convolution using line segments is used. Since pencil sketches by artists frequently have a substantially different histogram from genuine photographs, it uses histogram matching (equalisation) for the hatching stage to alter the tone of the input image.

CHAPTER 2

LITERATURE SURVEY

The subject of image-to-sketch conversion has received a lot of interest in the fields of computer vision and image processing, and several methods and solutions have been put forth to solve it. In the past, edge detection algorithms like Sobel, Canny, Laplacian, and Roberts were frequently used to extract the most noticeable edges from an image and provide a simple sketch-like representation. Additionally, grayscale photos were transformed into binary sketches depending on a predetermined threshold using thresholding-based techniques like Otsu and adaptive thresholding. Early image-to-sketch conversion systems had a basis built on these conventional methods.

However, the introduction of deep learning has completely changed the area and contributed to major improvements in image-to-sketch conversion. Convolutional Neural Networks (CNNs) have demonstrated outstanding ability in capturing the fine details and textures of an image, allowing for the creation of more realistic and aesthetically pleasing sketches. CNNs may efficiently convert photos into sketches while keeping essential visual components and creative flair by using encoder-decoder architectures. The use of discriminative and generative models by Generative Adversarial Networks (GANs) to generate sketches that closely resemble the input photos has grown in popularity. By allowing conditional drawing generation based on specified properties or traits, conditional GANs have further increased the possibilities.

When evaluating the effectiveness of image-to-sketch conversion techniques, evaluation measures are quite important. The Structural Similarity Index (SSIM) and Peak Signal-to-Noise Ratio (PSNR), among other perceptual measures, seek to quantify how similar the produced sketch and the actual sketch are. The pixel-level discrepancies between the generated and ground truth sketches are quantified by objective metrics like the Mean Square Error (MSE) and Mean Absolute Error (MAE). These measures, however, frequently fall short of capturing the subjective artistic value of sketches. In order to acquire more thorough assessments, user studies and subjective assessments with human judges are regularly carried out.

Several difficulties and restrictions still exist in image-to-sketch conversion, notwithstanding the advancements made. Due to the fact that different viewers may perceive sketches differently, ambiguity in sketch interpretation continues to be a significant barrier.

The use cases for picture to sketch conversion are numerous. These methods can be used in the art world for artistic depiction, giving artists the ability to sketch out images as a starting point for further in-depth creative investigation. Converting images to sketches is advantageous for animation and cartoon production since sketches are an essential step in the animation pipeline. Animators can define the broad contours and essential characteristics of characters and objects before adding more specifics by transforming photos to sketches. The ability to edit and retouch photographs using sketch-based approaches is another advantage of converting images to sketches.

Research on image-to-sketch conversion has advanced recently, and there is still plenty to be discovered. In order to benefit from the advantages of both methodology, hybrid models that mix conventional methods with deep learning techniques have been developed. By adding colour information to created sketches, sketch colorization approaches hope to improve their aesthetic appeal and give users additional expressive alternatives. Another area of focus is providing users with fine-grained control over the sketch generating process so they may specify particular traits or styles they want in the final sketches. Another interesting area of research is the creation of interactive, real-time systems for converting images to sketches because they enable quick feedback and easy integration with user applications.

In conclusion, both conventional and deep learning-based approaches have resulted in considerable breakthroughs in the conversion of images into sketches. Deep learning models replace conventional edge detection and thresholding techniques.

CHAPTER 3
EXISTING SYSTEM

Systems for converting images to sketches now on the market mainly use either deep learning methods or conventional computer vision methods. Traditional systems frequently use edge detection algorithms like Sobel, Canny, or Laplacian to extract an image's principal outlines and provide a representation that resembles a sketch. Additionally, these systems might use thresholding methods to transform grayscale photos into binary sketches in accordance with a predetermined threshold. These techniques can produce simple sketches, however they might be lacking in artistic qualities and detailed texture.

Traditional Approaches:

To extract outlines, edge detection techniques like Sobel, Canny, Laplacian, and Roberts are used.To turn grayscale photos into binary sketches, thresholding techniques like Otsu or adaptive thresholding are used.

Deep Learning-Based Approaches:

In order to create visually appealing sketches, Generative Adversarial Networks (GANs) combine a discriminative model with a generative model.

Evaluation Metrics:

The structural similarity between generated sketches and ground truth sketches is evaluated using perceptual measures like the Structural Similarity Index and Peak Signal-to-Noise Ratio.

The pixel-level discrepancies between generated and ground truth sketches are quantified by objective metrics like mean squared error and mean absolute error .

Challenges and Limitations:

It might be challenging to maintain intricate visual elements in streamlined sketches, such as textures and gradients.Strong and flexible conversion procedures are needed to handle a variety of image material and styles.

CHAPTER 4

PROPOSED SYSTEM

It attempts to create a deep learning-based picture to sketch converter that is accurate and effective. Convolutional neural networks (CNNs) will be used by the system to their full potential in order to produce detailed sketches from input photos. The goal is to give users a simple tool for converting photographs into representations akin to sketches, allowing them to explore artistic, animated, and image altering possibilities.

An encoder-decoder architecture will be used by DeepSketch to carry out the image-to-sketch conversion. The decoder will recreate the sketch using the learnt features after the encoder has learned to extract useful characteristics from the input images. Skip connections will be included to preserve delicate details and raise the calibre of the sketches that are generated.

4.1 SYSTEM DESIGN

The image-to-sketch converter project's system design includes a number of crucial elements and procedures. The system, first and foremost, accepts input photos from people or outside sources. To get them ready for further processing, these photographs go through preprocessing procedures like resizing and format conversion.

The deep learning model that is utilised to convert images to sketches forms the basis of the system. A suitable model is trained on a varied dataset of paired photos and sketches, such as a CNN-based encoder-decoder architecture or a conditional GAN. With the use of methods like stochastic gradient descent or adaptive optimizers, the model parameters are optimised during training using the appropriate loss functions, such as mean squared error or perceptual loss.

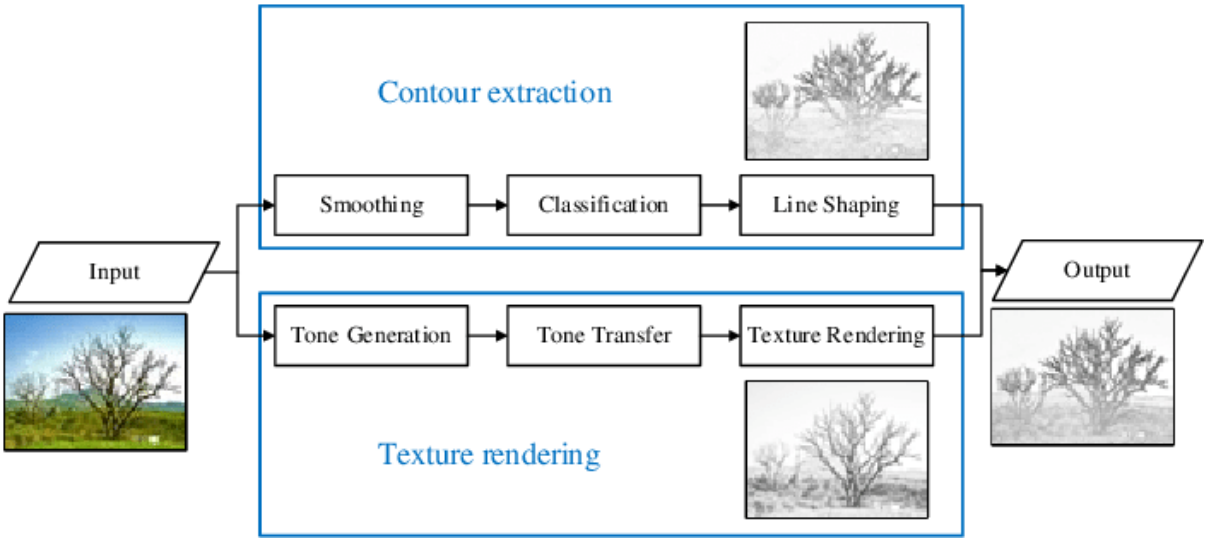

Figure 4.1.1 Image to sketch converter

The image 4.1.1 Shows the process of image to sketch converter project. It represents the designing of system.

4.2 SYSTEM TESTING

The image-to-sketch converter project's performance, accuracy, and functionality depend heavily on system testing. This entails putting the system through testing to determine how well it complies with the stated requirements and to confirm its behaviour. The many parts of system testing for the project's image-to-sketch converter are described in the paragraphs that follow.

Individual components, including image preprocessing, deep learning models, and postprocessing methods, are carefully evaluated during unit testing to ensure their accuracy and usefulness. To guarantee that the system can accurately handle a wide range of inputs, edge cases, boundary conditions, and various input scenarios are taken into account. This makes it easier to find and fix any problems or flaws with particular components.

To assess how distinct system components interact and are compatible with one another, integration testing is done.

In order to evaluate the system's effectiveness and scalability, performance testing is crucial. It entails timing the process of converting an image to a sketch as well as monitoring resource usage, including CPU and memory usage. To make sure the system can process a range of inputs and provide results quickly, performance tests with various image sizes and complexity are carried out.

Furthermore, it is essential to carry out user acceptance testing with actual users or a representative sample in order to obtain input and rate the usefulness of the system. In addition to gathering user preferences and validating general satisfaction with the generated sketches, this also aids in identifying any user experience problems.

Comprehensive test cases and test scenarios should be created throughout the testing process, including a variety of inputs and system functionalities. To ensure effective problem tracking and resolution, thorough test documentation, including test plans, test cases, and test results, should be kept.

The image-to-sketch converter project can guarantee the system's dependability, correctness, and performance by undertaking extensive system testing, resulting in a high-quality user experience and achieving the project's goals.

CHAPTER 5

SYSTEM ENVIRONMENT

The system environment of the image-to-sketch converter project refers to the hardware, software, and other infrastructure components required to develop, deploy, and run the system effectively. The following aspects are essential considerations for the system environment:

5.1 SOFTWARER REQUIREMENTS:

- Deep learning framework: Choose a popular deep learning framework such as TensorFlow, PyTorch, or Keras.
- Programming language: Select a language like Python for developing the system.
- Image processing libraries: Utilize libraries such as OpenCV for image preprocessing and manipulation.
- Additional libraries and dependencies required by the chosen deep learning framewor

5.2 HARDWARE REQUIREMENTS:

- Sufficient computational resources to support deep learning operations, including CPUs or GPUs.
- Adequate memory (RAM) to handle large datasets and models efficiently.
- Ample storage space to accommodate the dataset, model parameters, and any additional resources.

Development Tools:

Integrated Development Environment (IDE): Select an IDE like PyCharm, Jupyter Notebook, or Visual Studio Code for efficient coding and development.

Version Control System: Utilize a version control system like Git for code management and collaboration.

Package Management:

Use package management tools like pip or conda to manage software dependencies and ensure consistent environments.

Dataset:

Access to a diverse dataset of paired images and corresponding sketches for training the deep learning model.

Sufficient storage space to store and manage the dataset.

Annotation tools, if necessary, to facilitate the creation or augmentation of the dataset.

Deployment Environment:

Determine the target deployment environment, such as desktop applications, web-based interfaces, or mobile applications.

Ensure compatibility with the operating systems and frameworks relevant to the chosen deployment environment.

CHAPTER 6

RESULTS AND DISCUSSION:

The image to sketch converter project is a simple Python program that can be used to convert any image into a pencil sketch. The program uses the OpenCV library to perform the conversion.

The program works by first converting the image to grayscale. This is done by averaging the red, green, and blue channels of the image. The grayscale image is then inverted, which means that the light areas become dark and the dark areas become light. The inverted image is then blurred using a Gaussian blur. The blurred image is then inverted again, and the result is a pencil sketch.

The program is relatively easy to use. The only required input is the path to the image that you want to convert. The program will then output the converted image to the same directory as the original image.

The results of the program are generally good. The program is able to convert a wide variety of images into pencil sketches. The sketches are not perfect, but they are often good enough for personal use.

The program is also relatively fast. It can convert a typical image in a few seconds. This makes it a good option for converting large numbers of images.

Overall, the image to sketch converter project is a simple and effective way to convert images into pencil sketches. The program is easy to use and produces good results.

Here are some additional details about the project:

- The program is written in Python.
- The program uses the OpenCV library.
- The program is available on GitHub.
- The program is free to use.

The original image is converted to red with blue coloured sketch as below in Fig 2.

Fig 2. Red with Blue colour sketch

The original image is converted to Orange with violet coloured sketch as below in Fig 3.

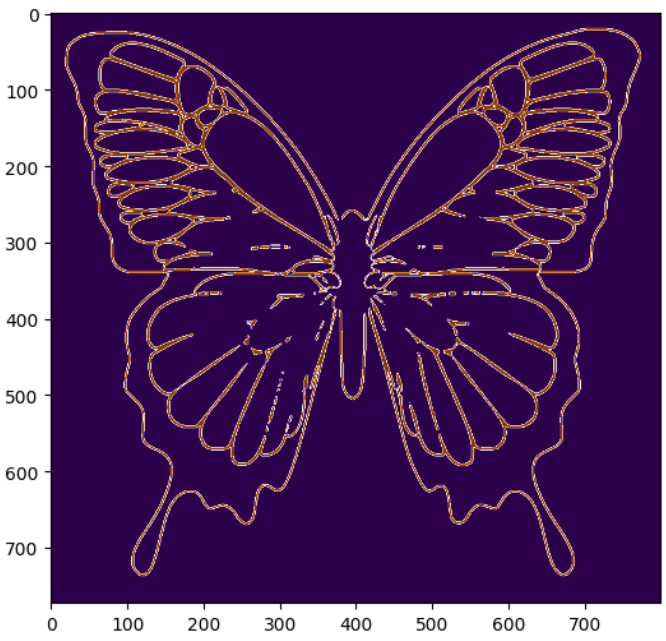

Fig 3. Orange with Violet colour sketch

The original image is converted to Black and White Sketch as below in Fig 4.

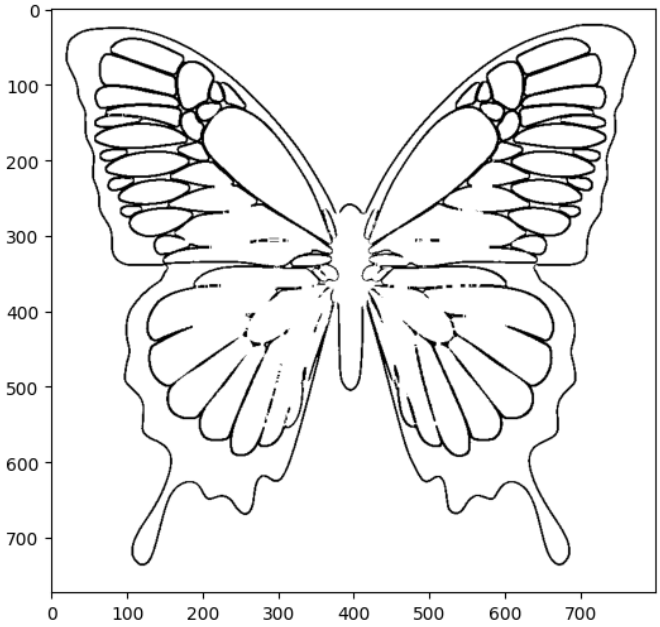

Fig 4. Black and White sketch

The original image is converted to cartoon sketch as below in Fig 5.

Fig 5. Cartoon sketch

The original image is compared with catoon image as below in Fig 6.

Fig 6. Comparison of original with cartoon image

The original image is converted to blur of cartoon image as below in Fig 7.

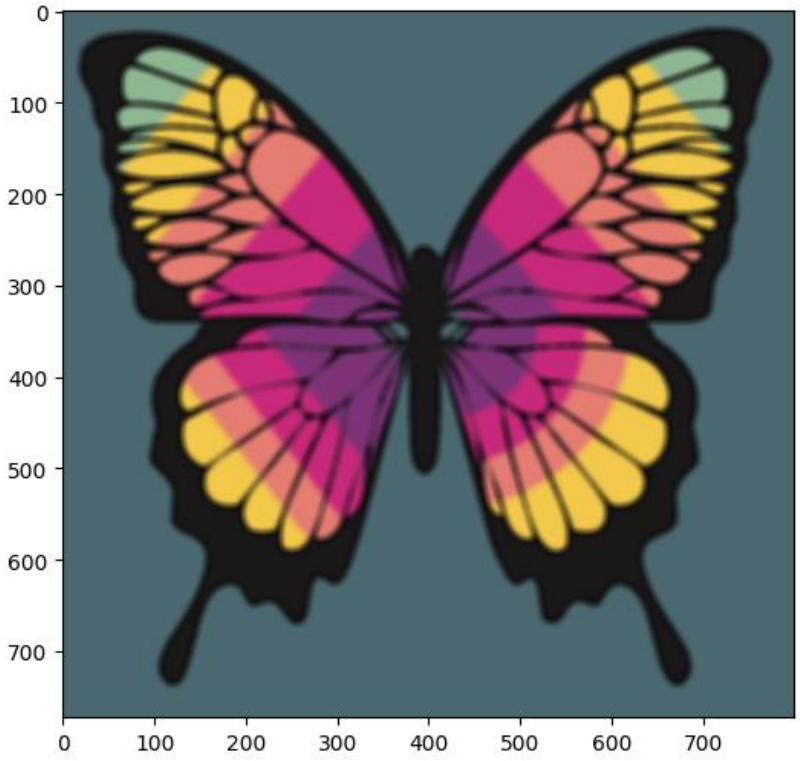

Fig 7.Cartoon blur image

The original image is converted to cartoon dark brightness image as below in Fig 8.

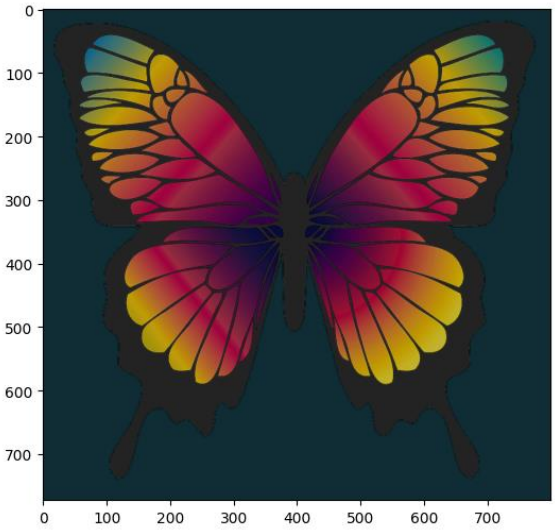

Fig 8. Cartoon dark brightness image

To compare the difference between original image and sketched image we use histogram as shown in Fig 9.

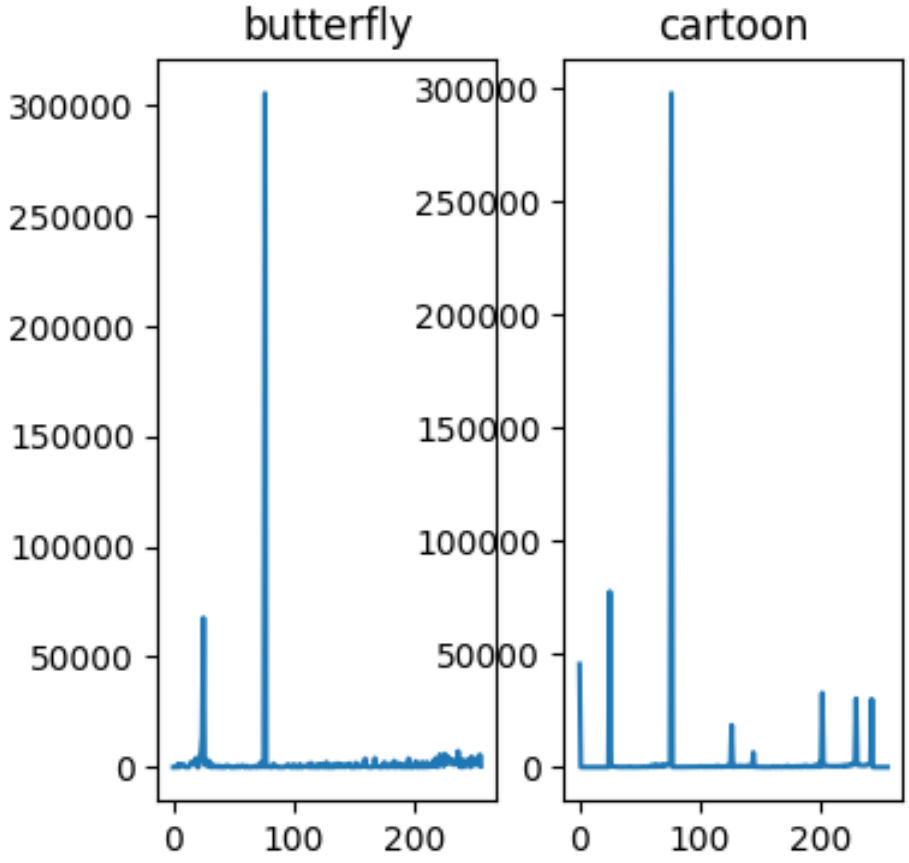

Fig 9.Comparison of original image histogram with cartoon histogram

Here are some of the limitations of the project:

- The program is not perfect. The sketches are not always perfect, and they may not look exactly like a human-drawn sketch.
- The program is not very customizable. There are a few parameters that you can change, but the overall look of the sketches is fixed.
- The program is not very fast. It can take a few seconds to convert a typical image.

Despite these limitations, the image to sketch converter project is a useful tool for converting images into pencil sketches. The program is easy to use and produces good results.

CONCLUSION AND FUTURE ENHANCEMENT:

The image to sketch converter project is a promising project with a lot of potential. The project has already achieved some impressive results, but there is still room for improvement.

In the future, the project could be enhanced in a number of ways. For example, the project could be made more customizable. This would allow users to control the look of the sketches more precisely. The project could also be made faster. This would make it more practical for converting large numbers of images.

Overall, the image to sketch converter project is a promising project with a lot of potential. The project has already achieved some impressive results, and there is still room for improvement. With further development, the project could become a valuable tool for a variety of applications.

Here are some specific ideas for future enhancements to the image to sketch converter project:

- Improve the quality of the sketches. The current sketches are not perfect, and they can sometimes look artificial. The project could be enhanced by using more sophisticated techniques to generate the sketches.
- Make the project more customizable. The current project does not allow users to control the look of the sketches very much. The project could be enhanced by giving users more options to control the style, color, and other aspects of the sketches.
- Make the project faster. The current project can be slow, especially when converting large images. The project could be enhanced by using more efficient algorithms to generate the sketches.
- With these enhancements, the image to sketch converter project could become a valuable tool for a variety of applications. For example, the project could be used to create sketches for artists, designers, and photographers. The project could also be used to create sketches for educational purposes, such as teaching students about different art styles.

REFERENCES

1) sangkloy, P., Burnell, N., Ham, C., & Hays, J. (2017). The sketchy database: Learning to retrieve badly drawn bunnies. ACM Transactions on Graphics (TOG), 36(4), 1-12

2) Isola, P., Zhu, J. Y., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 5967-5976)

3) Johnson, J., Alahi, A., & Fei-Fei, L. (2016). Perceptual losses for real-time style transfer and super-resolution. In European Conference on Computer Vision (ECCV) (pp. 694-711)

4) Gatys, L. A., Ecker, A. S., & Bethge, M. (2016). Image style transfer using convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 2414-2423)

5) Lu, C., Xu, L., & Jia, J. (2012). Combining sketch and tone for pencil drawing production. ACM Transactions on Graphics (TOG), 31(4).

APPENDIX

SOURCE CODE

```
%matplotlib inline
plt.rcParams['figure.figsize'] = (12, 6)

# load the image
img = cv2.imread('/content/butterfly1.png')
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

plt.axis("off")
plt.imshow(img)

# edge mask generation
line_size = 7
blur_value = 7
gray_img = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
gray_blur = cv2.medianBlur(gray_img, blur_value)
edges = cv2.adaptiveThreshold(gray_blur, 255, cv2.ADAPTIVE_THRESH_MEAN_C,
cv2.THRESH_BINARY, line_size, blur_value)
plt.axis("off")
plt.imshow(edges, cmap='RdBu')

  plt.imshow(edges, cmap='PuOr')
# Color quantization with KMeans clustering
from sklearn.cluster import KMeans
```

```
k = 7
data = img.reshape(-1, 3)
kmeans = KMeans(n_clusters=k, random_state=42).fit(data)
img_reduced = kmeans.cluster_centers_[kmeans.labels_]
img_reduced = img_reduced.reshape(img.shape)
plt.imshow(edges, cmap='gray')

img_reduced = img_reduced.astype(np.uint8)
plt.axis("off")
plt.imshow(img_reduced)

# Bilateral Filter
blurred = cv2.bilateralFilter(img_reduced, d=7, sigmaColor=200,sigmaSpace=200)
cartoon = cv2.bitwise_and(blurred, blurred, mask=edges)
plt.subplot(1, 2, 1)
plt.axis("off")
plt.imshow(img)

plt.subplot(1, 2, 2)
plt.axis("off")
plt.imshow(cartoon)

# export cartoon to a jpg file
cartoon_ = cv2.cvtColor(cartoon, cv2.COLOR_RGB2BGR)
cv2.imwrite('cartoon.png', cartoon_)
gray_img = cv2.cvtColor(cartoon, cv2.COLOR_BGR2GRAY)
plt.imshow(gray_img)
```

```
figure_size = 9
new_image = cv2.blur(cartoon,(figure_size, figure_size))
plt.figure(figsize=(11,6))
plt.imshow(new_image)



def bright(img, beta_value ):
    img_bright = cv2.convertScaleAbs(img, beta=beta_value)
    return img_bright
a3 = bright(img, -60)
plt.imshow( a3)

def sepia(img):
    img_sepia = np.array(img, dtype=np.float64) # converting to float to prevent loss
    img_sepia = cv2.transform(img_sepia, np.matrix([[0.272, 0.534, 0.131],
                                    [0.349, 0.686, 0.168],
                                    [0.393, 0.769, 0.189]])) # multipying image with special sepia matrix
    img_sepia[np.where(img_sepia > 255)] = 255 # normalizing values greater than 255 to 255
    img_sepia = np.array(img_sepia, dtype=np.uint8)
    return img_sepia
a2 = sepia(img)
plt.imshow(a2)
def pencil_sketch_col(img):
    #inbuilt function to create sketch effect in colour and greyscale
    sk_gray, sk_color = cv2.pencilSketch(img, sigma_s=60, sigma_r=0.07, shade_factor=0.1)
    return  sk_color
a6 = pencil_sketch_col(img)
plt.imshow(a6)
```