

Image Caption Generation with Audio using Deep Learning Techniques

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering Degree in Computer Science and
Engineering

By

M.V.SESHAN
(Reg. No. 38110516)
And

SACHIN MUDALIAR
(Reg. No. 38110473)



**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING SCHOOL OF COMPUTING**

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

MAY - 2022



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

[DEEMED TO BE UNIVERSITY]

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **SACHIN MUDALIAR (38110473)** and **M.V.SESHAN (38110516)** who carried out the project entitled "**Image Caption Generation with Audio using Deep Learning Techniques**" under my supervision from November 2021 to April 2022.

Internal Guide

Dr. G. Kalaiarasi

Head of the Department

Dr. S. VIGNESHWARI, M.E., Ph.D.,

Submitted for Viva voce Examination held on

Internal Examiner

External Examiner

DECLARATION

I **SACHIN MUDALIAR (38110473)** hereby declare that the Project Report entitled **Image Caption Generation with Audio using Deep Learning Techniques** done by me under the guidance of **Dr. G. Kalaiarasi** (Internal) at **Sathyabama Institute of Science and Technology, CHENNAI** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE: 10.04.2022

Sachin Mudaliar



PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D., Dean**, School of Computing , **Dr.S.Vigneshwari M.E., Ph.D.**, and **Dr.L.Lakshmanan M.E., Ph.D.**, Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. G. Kalaiarasi** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

Sachin Mudaliar

(38110473)

CSE

ABSTRACT

The convolution neural network method is among the most widely used deep learning-based object detection methods. This system mainly focuses on image captioning based on research papers. Different Captioning metrics are used for evaluation of the sentences generated by the system. The scores tell about the accuracy of the words obtained. Different methods are compared which tell the efficiency of the LSTM method to be 80%. This provides best results on Visual Genome Dataset. The output generated can have few limitations i.e, the paragraph can contain upto 500 words or 4-5 lines. Hence, this system provides a clear view of how a paragraph is generated from an image. The scope of the paper is limited to LSTM approach only. In future, the scope of the work can be extended so that the system can be more efficiently used by all the researchers. In order to address these problems, a new cluster method for estimating the initial width and height of the predicted bounding boxes has been developed.

INTRODUCTION

Consistently a great deal of photographs is received from the climate, web-based media and magazines. One can know the photograph. As people, it can decided to take photographs without clarifying them, however then again, machines need a preparation picture, and afterward give a composed photograph. Video depictions can be helpful for an assortment of reasons, for instance, supporting the visually impaired through informal utilizing genuine plans to envision the condition of the camera, and advancing general wellbeing by changing over photograph sharing to a public site. furthermore, instant messages. Assist youngsters with mastering things and foster language abilities. The data on every photograph on the world's web is quick and permits you to look for and show definite photographs. Picture depictions come in different bundles in biomedicine, trade, online pursuit, boats and considerably more. It is feasible to remark on photographs on open locales like Instagram and Facebook, and the principle reason for this review is to have less insight and a more profound learning style. This paper utilizes two systems to make pictures: CNN and LSTM.

OBJECTIVE

The objective of our project is to learn the concepts of a CNN and LSTM model and build a working model of Image caption generator by implementing CNN with LSTM.

In this Python project, we will be implementing the caption generator using *CNN (Convolutional Neural Networks)* and LSTM (Long short term memory). The image features will be extracted from Xception which is a CNN model trained on the Flickr8k dataset and then we feed the features into the LSTM model which will be responsible for generating the image captions.

Literature survey

Title 1: Camera2Caption: A real-time image caption generator (2017)

Author: Pranay Mathur, Aman Gill, Aayush Yadav, Anurag Mishra, Nand Kumar Bansode

Proposed methodology:

Proposed Deep Learning Based Advanced Technique Deep Reinforcement Learning that's led by Computer Vision and machines translation based on deep learning model. Dataset used in this model is MS-COCO.

Conclusions:

The proposed model based on deep learning, well optimize and perform in real time environment (mobile devices) and produce high quality captions by using help of tensorflow by google.

Title 2: Image Captioning: Transforming Objects into Words. June 2019

Author: Simao Herdade, Armin Kappeler, Kofi Boakye, Joao Soares

Proposed Object Relation Transformer model, focuses on spatial relationship between objects of images through used of faster R-CNN with ResNet101. Mainly focuses on Improve the relationship between objects. Dataset used in this model is MS-COCO with Pycharm IDE

Conclusion: The proposed model encodes positions and size and relationship between detected objects in images and extracted features by building upon the bottom-up and topdown image captioning approach and CNN.

Title 3: Automatic Image Captioning Using Convolution Neural Networks and LSTM November 2019

Author: R subhash

Proposed Deep Learning based Convolution Neural Networks and Natural Language Processing (NLP) Techniques reasonable sentences are framed and inscriptions are produced. Dataset used in this model is MS-COCO.

Conclusion: Proposed model having convolution neural network whose output is paired to Long Short Term Memory network which helps us generate descriptive captions for the image. Also model don't require huge dataset to produce caption of images.

Title 4: Image Caption Generator Using Deep Learning

Author: B.Krishnakumar,K.Kous alya, S.Gokul,R.Karthikeyan, D.Kaviyarasu

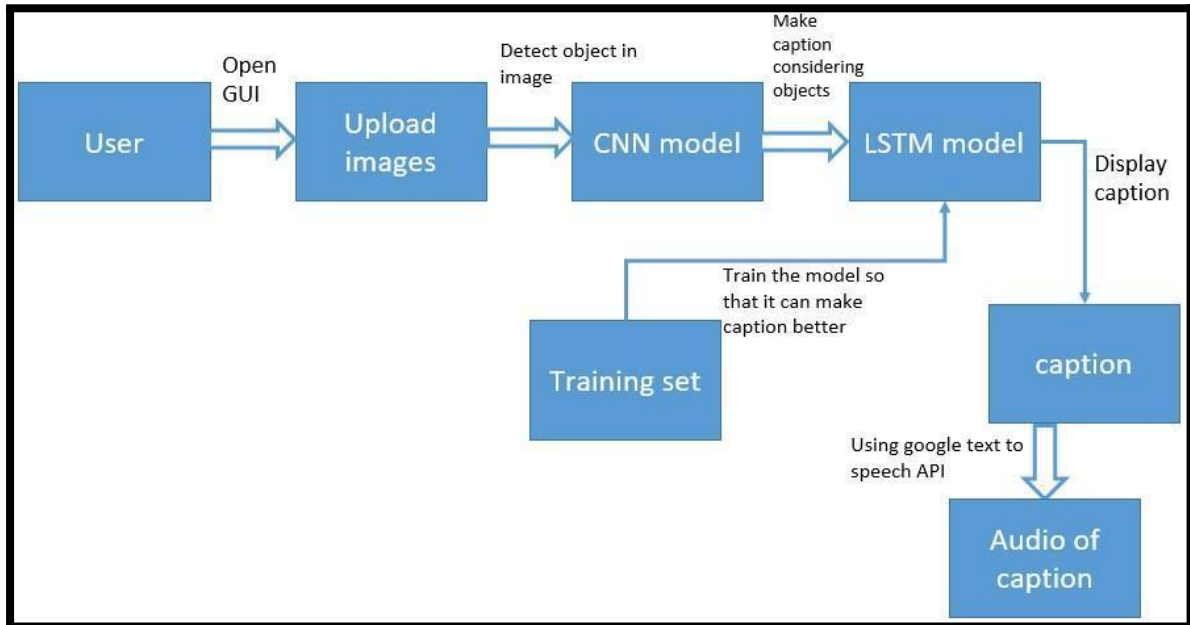
Proposed Deep Learning based Convolution Neural Networks to identify objects in the images using OpenCv. Detected Images converted into audio using GTTP and then converted to text using Long Short Term Memory network.They used Pre-trained model VGG16 as a baseline model

Conclusion: Proposed Model successfully trained to generate captions of images using CNN technique, model is depends on data and used small data set. The model generate caption by using Keras Framework used in Jupyter notebook and also conclude keras has strong support for multiple GPU's.

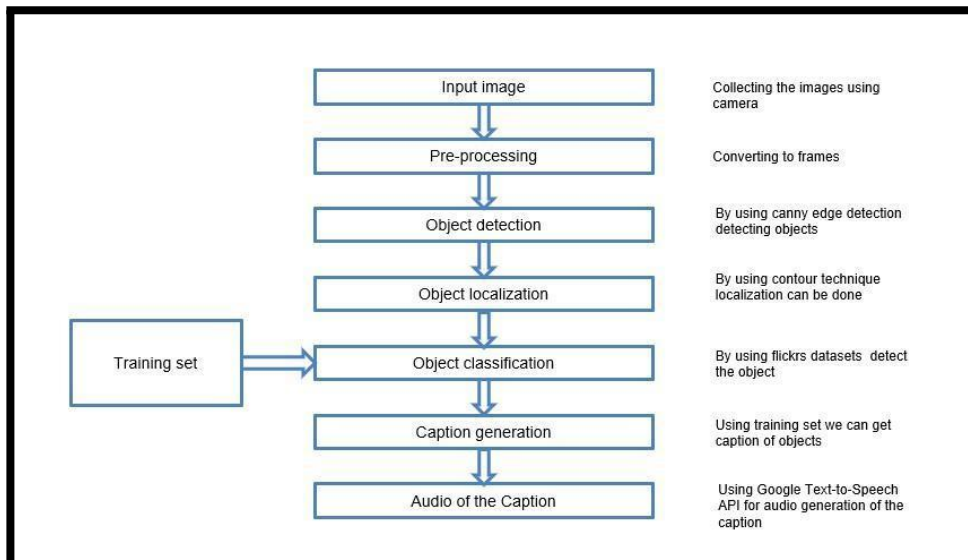
Proposed system

- This paper proposes a new method for initializing the width and height of predicted bounding boxes. Our proposed method has a larger Avg IOU and smaller running time on the MS COCO dataset.
- The Avg IOU is 60.44%, which is 0.56% higher than original YOLOv3 method, and the running time is 1/297 that of the original YOLOv3 method. For the PASCAL VOC dataset, the average IOU is 67.45%, which is 0.13% higher than original YOLOv3 method, and the running time is 1/81 that of the original YOLOv3 method.
- It exhibits better performance in terms of initializing the width and height of predicted bounding boxes, as well as in terms of choosing the representative initial width and height. Besides, we randomly selected some images from the test set of the MS COCO dataset for detection.
- The object detection results indicate that our proposed method detected more objects in some test images. It also has better performance in terms of detecting small objects. Our proposed method also outperforms the original YOLOv3 method in terms of recall, mean average precision, and F1-score.

Architecture diagram



Flow chart



Models

1. Collect the datasets

- From the camera we can get video, and some data set collecting in the form images.

2. Feature extraction

- Applying edge and localization on the images.

3. Applying algorithm

- YOLOv3 and MobileNet algorithms are applied.

4. Classification analysis

- Object can be classified.

DIGITAL IMAGE PROCESSING

Digital Image Processing :-

- Digital Image Processing is a software which is used in image processing. For example: computer graphics, signals, photography, camera mechanism, pixels, etc.
- Digital Image Processing provides a platform to perform various operations like image enhancing, processing of analog and digital signals, image signals, voice signals etc.
- It provides images in different formats.

Digital Image Processing allows users the following tasks :-

- **Image sharpening and restoration:** The common applications of Image sharpening and restoration are zooming, blurring, sharpening, grayscale conversion, edges detecting, Image recognition, and Image retrieval, etc.
- **Medical field:** The common applications of medical field are Gamma-ray imaging, PET scan, X-Ray Imaging, Medical CT, UV imaging, etc.
- **Remote sensing:** It is the process of scanning the earth by the use of satellite and acknowledges all activities of space.
- **Machine/Robot vision:** It works on the vision of robots so that they can see things, identify them, etc.

Characteristics of Digital Image Processing :-

- It uses software, and some are free of cost.
- It provides clear images.
- Digital Image Processing do image enhancement to recollect the data through images.
- It is used widely everywhere in many fields.
- It reduces the complexity of digital image processing.
- It is used to support a better experience of life.

Advantages of Digital Image Processing :-

- Image reconstruction (CT, MRI, SPECT, PET)
- Image reformatting (Multi-plane, multi-view reconstructions)
- Fast image storage and retrieval
- Fast and high-quality image distribution.
- Controlled viewing (windowing, zooming)

Disadvantages of Digital Image Processing :-

- It is very much time-consuming.
- It is very much costly depending on the particular system.
- Qualified persons can be used.

History of Photography :-

The first process of photography was called **heliography** which was invented by **Nicephore Niepce** in the year **1824**. In this process, images are obtained after several days by bitumen of Judea on a silver plate.

In 1829, Niépce relates to Louis Jacques Mandé Daguerre in his research.

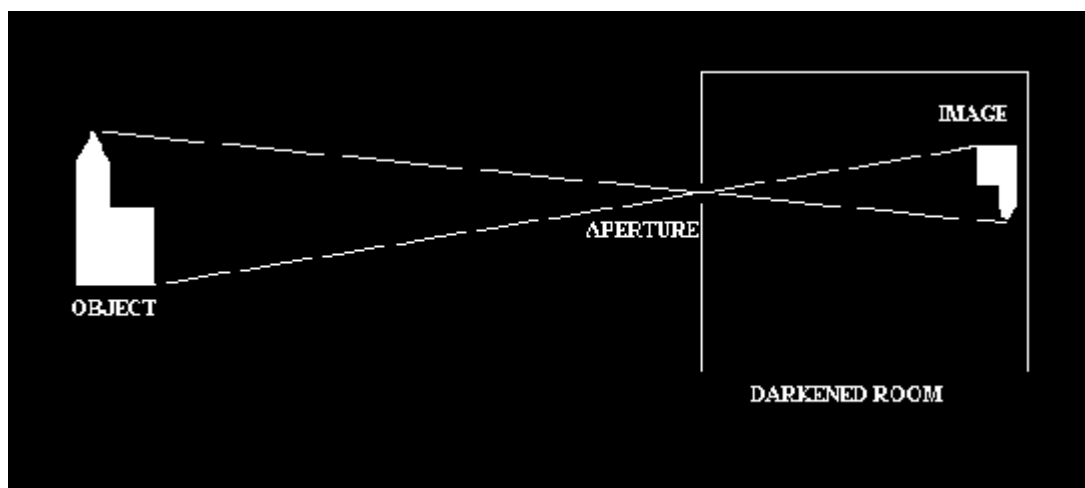
Origin of the camera :-

There were several scientific inventions that took place for the betterment of people. One of the best achievement was the invention of the camera which can catch human activity. Photographic technology is changed over many generations.

It starts with camera obscura and then extends with the introduction of new technology including **calotypes, dry plates, daguerreotypes, and today digital cameras**. Digital cameras have completely changed the history of photography because it produces the best quality of images.

Camera Obscura :-

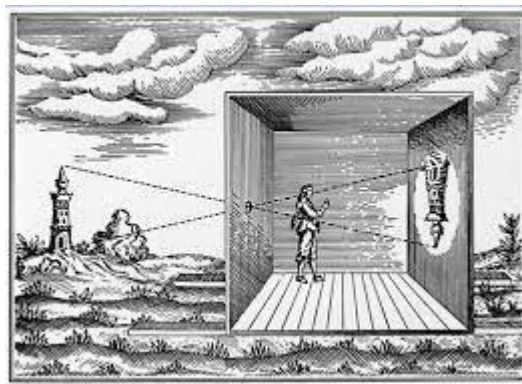
In Latin word, the Obscura means a dark room, which was used to take the picture from the other side of the camera. This concept was taken from a Chinese concept to where images are projected on a surrounding wall.



an example of camera obscura

Working of camera obscura

It was an ancient optical device which was used to capture an image. It was the most basic form of a camera which was quite simple. In a dark room, a small hole was placed on the opposite wall such that whatever be outside, was appeared on the wall but the image captured was upside-down or inverted.



an image generated by camera obscura

While capturing the images, the size of the hole present in the darkroom played a very important role. If the size of the hole was small, then a sharp and well-focused image had appeared on the wall and if the size of the hole was larger, then a bright image was produced but that was very less focused.

Due to the rectilinear propagation of light, the light travelled in a straight line. Later on, **Vermeer and Canaletto** two artists worked on brightness and clarity of an image.

Portable Cameras vs. Digital Cameras :-

The man who builds first portable camera was **Mr. Johann Zahn**. In initial days, the size of cameras was big and heavy too. So, it becomes impossible to take it anywhere. In the late 19th century, the portable cameras were introduced by **Mr. Robert Boyle and Robert Hooke**. It was a portable camera but was heavy in size. So, they work further to find the ways that they can resize the camera.

Origin of photography :-

In the 19th century camera, Obscura was introduced by a scientist but its actual use was done by **Mr. Roger Bacon**, an English philosopher to catch the solar eclipses pictures.

Portable camera :-

Today, there is a long list of portable cameras but if we talk about old ones then we come to know that it works on the principle of a pin hole camera. In a very dark room, images of illuminated objects penetrate through a very small hole. On the opposite wall, you will see the objects, in its proper form and colour. In a reversed position, its size got reduced due to the intersection of rays.

Examples of portable cameras are Leica ad Argus, analog CCTV cameras, etc.

Digital camera :-

In 1991, Sony Company launched their first digital camera that was Mavica, a magnetic video camera. In this camera, the images were stored in floppy disks and later seen on any computer screen. It was not a fully digital camera but

due to its image storing capacity, it comes in highlight and it had a 0.3 megapixel capacity of capturing photos.

Later on, with the changing technology new high pixels camera got introduced and they come under a fully digital category.

Features of digital camera :-

- Resolution of the camera is very high
- Maximum Lens aperture
- Lens zoom range
- High Lens quality
- In-camera Software

A large number of the applicator are governed under digital image processing. Almost in every field DIP is using in order to get the images.

Applications of Digital Image Processing :-

Almost in every field, digital image processing puts a live effect on things and is growing with time to time and with new technologies.

1) Image sharpening and restoration

It refers to the process in which we can modify the look and feel of an image. It basically manipulates the images and achieves the desired output. It includes conversion, sharpening, blurring, detecting edges, retrieval, and recognition of images.

2) Medical Field

There are several applications under medical field which depends on the functioning of digital image processing.

- Gamma-ray imaging
- PET scan
- X-Ray Imaging
- Medical CT scan
- UV imaging

3) Robot vision

There are several robotic machines which work on the digital image processing. Through image processing technique robot finds their ways, for example, hurdle detection robot and line follower robot.

4) Pattern recognition

It involves the study of image processing, it is also combined with artificial intelligence such that computer-aided diagnosis, handwriting recognition and images recognition can be easily implemented. Now a days, image processing is used for pattern recognition.

5) Video processing

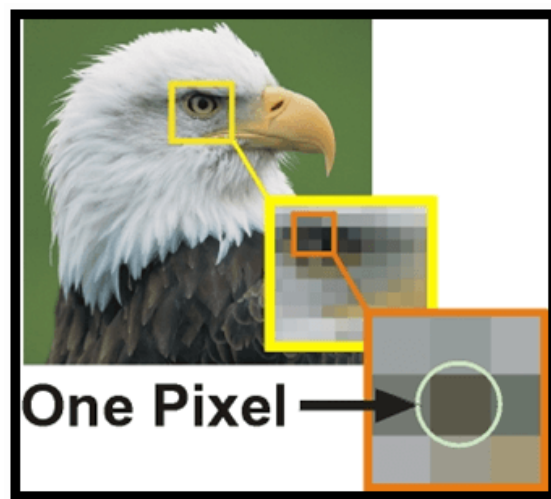
It is also one of the applications of digital image processing. A collection of frames or pictures are arranged in such a way that it makes the fast movement of pictures. It involves frame rate conversion, motion detection, reduction of noise and colour space conversion etc.

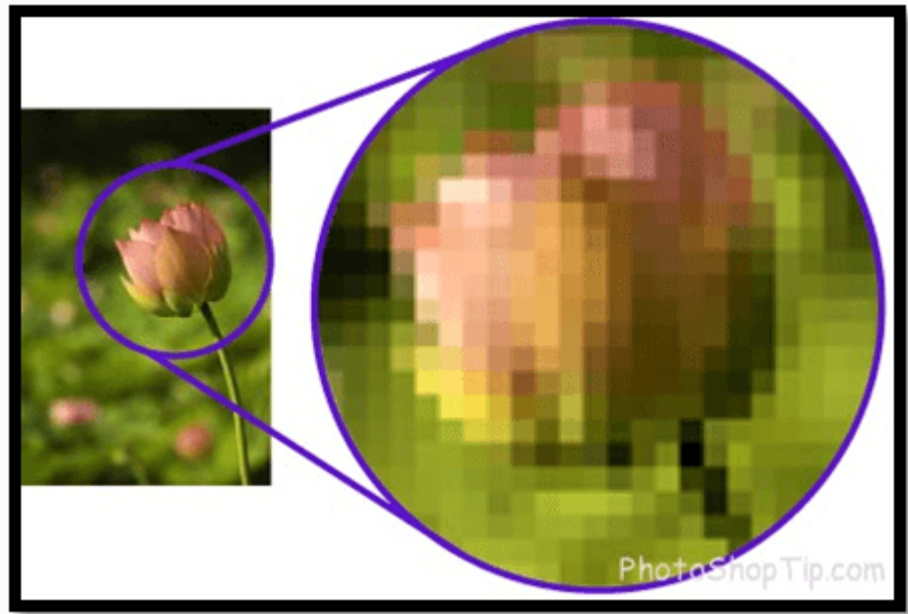
Concept of Pixel :-

The full form of the pixel is "Picture Element." It is also known as "**PEL.**" Pixel is the smallest element of an image on a computer display, whether they are LCD or CRT monitors. A screen is made up of a matrix of thousands or millions of pixels. A pixel is represented with a dot or a square on a computer screen.



The good thing is that a pixel cannot be seen as they are very small which result in a smooth and clear image rather than "pixelated." Each pixel has a value, or we can say a unique logical address. It can have only one color at a time. Colour of a pixel is determined by the number of bits which is used to represent it. A resolution of a computer screen depends upon graphics card and display monitor, the quantity, size and color combination of pixels.





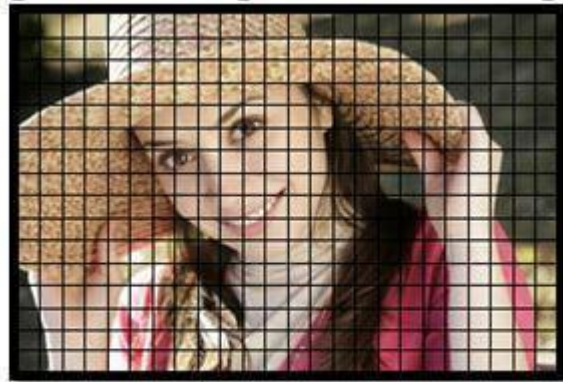
As we know that an image is build up of thousands and millions of pixels. In the above images, if we zoom in the image, we will be able to see some of the pixels.

Relationship with CCD array :-

When an image has zoomed in, the surface of CCD will look like filled dots. These dots are light receptor called photodiode.

$$\boxed{1 \text{ Dot}} = \boxed{1 \text{ Pixel}}$$

CCD sizes are described using Terms like 2 million pixels (megapixel) and 4 million pixels (megapixel). The more the count of pixel, more detailed image is generated. To get a clear and image smooth, CCD and image size is increased.



A Small number of CCD pixels



A Large number of CCD pixels

Calculation of the total number of pixels :-

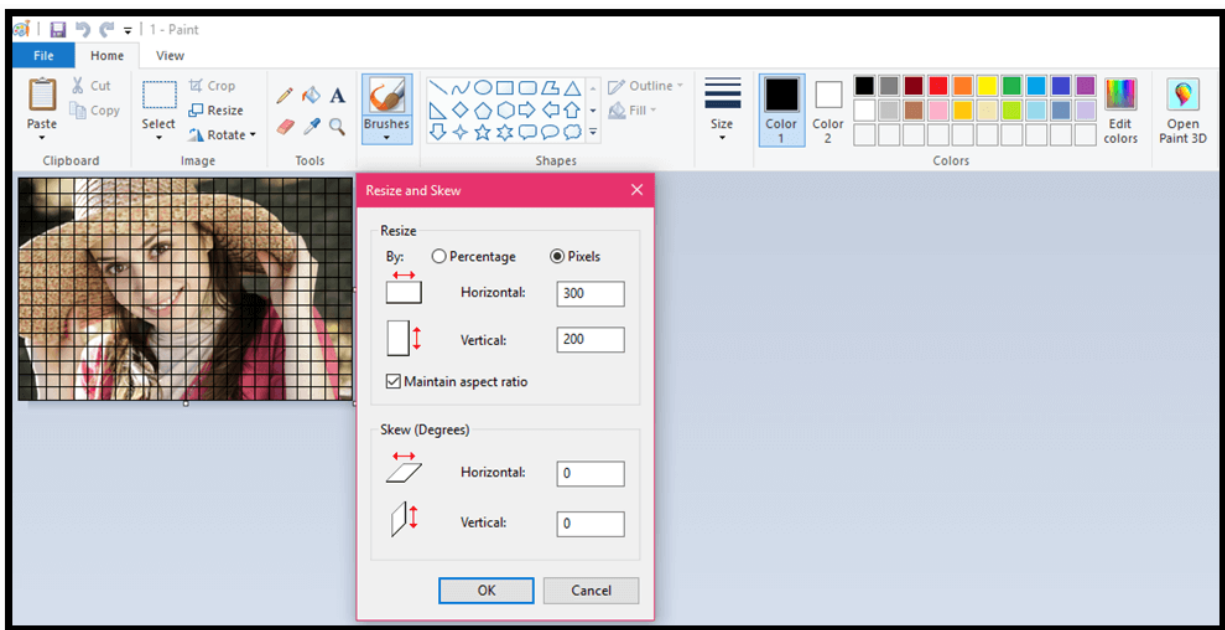
Below is the formula to calculate the total number of pixel in an image.



For example: let rows=300 & columns=200

Total number of pixels= 300 X 200

= 500



Gray level :-

The value of the minimum gray level is 0. The gray level depends on the depth of the image.

For example: In an 8-bit image, gray level is 255. For a binary image, a pixel can only take value 0 or 255. In color image, it can choose values between 0 and 255.

The formula for calculating gray level in a color image is as shown below:

$$\text{Grey level} = 0.299 * \text{red component} + 0.587 * \text{green component} + 0.114 * \text{blue component}$$

Pixel value(0)

As we know that each pixel has a unique value. 0 is a unique value that means the absence of light. It means that 0 is used to denote dark.

For example:

We have a matrix of 3X3 of an image, and each pixel is of value as shown below:

0	0	0
0	0	0
0	0	0

Total number of pixels

=

Number of rows

X

Number of columns

$$= 3 \times 3$$

$$= 9.$$

It means the image formed is of 9 pixels which are black.

Grayscale to RGB Conversion :-

To convert a grayscale image into color or RGB format, we have two methods. Both methods have their advantages as well as disadvantages.

Methods for conversion of grayscale image to RGB are as following:

1. Average method

It is the simplest method. We have to take an average of all the 3 colors.

Formula: $(R + G + B)/3$

For example:



RGB image



Grayscale image

Explanation

As we can see, there are changes in the image by applying the average method. But the result is unexpected as we want a grayscale image, but it turned to be a black image.

Problem

This problem occurs because we have taken an average of 3 colors. All the 3 colors have different wavelength and have their contribution to the formation of an image. In the above image, we are taking 33% from each portion that is why the image does not occur in grayscale.

The solution to this problem is given by the weighted method.

Weighted method or luminosity method

As we have seen the problem in the average method. And for this problem weight method is the solution. In all the 3 portions of color, red color has more wavelength, and green color has less wavelength. But green color gives a more soothing effect to the eyes as compared to red color.

By decreasing the value of red color and increasing the value of green color and the value for blue color will be between these two colors.

The equation for a new grayscale image will be:

Let the new grayscale image be X

$$X = (0.3 * R) + (0.59 * G) + (0.11 * B)$$

According to the above equation, red is used 30%, green is used 59%, and blue is used 11%. The contribution of green is highest.

By Applying the equation in an image, we will get:



Original Image



Grayscale Image

Explanation

As we can see using the weighted method the image is properly converted into grayscale. Image is brighter as compared with the image which was generated using the average method.

Sampling and quantization

In order to become suitable for digital processing, an image function $f(x,y)$ must be digitized both spatially and in amplitude. Typically, a frame grabber or digitizer is used to sample and quantize the analogue video signal. Hence in order to create an image which is digital, we need to convert continuous data into digital form. There are two steps in which it is done:

- Sampling
- Quantization

The sampling rate determines the spatial resolution of the digitized image, while the quantization level determines the number of grey levels in the digitized image. A magnitude of the sampled image is expressed as a digital value in image processing. The transition between continuous values of the image function and its digital equivalent is called quantization.

The number of quantization levels should be high enough for human perception of fine shading details in the image. The occurrence of false contours is the main problem in image which has been quantized with insufficient brightness levels.

Resizing image

Image interpolation occurs when you resize or distort your image from one pixel grid to another. Image resizing is necessary when you need to increase or decrease the total number of pixels, whereas remapping can occur when you

are correcting for lens distortion or rotating an image. Zooming refers to increase the quantity of pixels, so that when you zoom an image, you will see more detail.

Interpolation works by using known data to estimate values at unknown points. Image interpolation works in two directions, and tries to achieve a best approximation of a pixel's intensity based on the values at surrounding pixels. Common interpolation algorithms can be grouped into two categories: adaptive and non-adaptive. Adaptive methods change depending on what they are interpolating, whereas non-adaptive methods treat all pixels equally. Non-adaptive algorithms include: nearest neighbor, bilinear, bicubic, spline, sinc, lanczos and others. Adaptive algorithms include many proprietary algorithms in licensed software such as: Qimage, PhotoZoom Pro and Genuine Fractals.

Many compact digital cameras can perform both an optical and a digital zoom. A camera performs an optical zoom by moving the zoom lens so that it increases the magnification of light. However, a digital zoom degrades quality by simply interpolating the image. Even though the photo with digital zoom contains the same number of pixels, the detail is clearly far less than with optical zoom.

Aliasing and image enhancement

Digital sampling of any signal, whether sound, digital photographs, or other, can result in apparent signals at frequencies well below anything present in the original. Aliasing occurs when a signal is sampled at a less than twice the highest frequency present in the signal. Signals at frequencies above half the sampling rate must be filtered out to avoid the creation of signals at frequencies not present in the original sound. Thus digital sound recording

equipment contains low-pass filters that remove any signals above half the sampling frequency.

Since a sampler is a linear system, then if an input is a sum of sinusoids, the output will be a sum of sampled sinusoids. This suggests that if the input contains no frequencies above the Nyquist frequency, then it will be possible to reconstruct each of the sinusoidal components from the samples. This is an intuitive statement of the Nyquist-Shannon sampling theorem.

Anti-aliasing is a process which attempts to minimize the appearance of aliased diagonal edges. Anti-aliasing gives the appearance of smoother edges and higher resolution. It works by taking into account how much an ideal edge overlaps adjacent pixels.

Image enhancement: contrast enhancement

1. Image enhancement techniques have been widely used in many applications of image processing where the subjective quality of images is important for human interpretation. Contrast is an important factor in any subjective evaluation of image quality. Contrast is created by the difference in luminance reflected from two adjacent surfaces. In other words, contrast is the difference in visual properties that makes an object distinguishable from other objects and the background. In visual perception, contrast is determined by the difference in the colour and brightness of the object with other objects. Our visual system is more sensitive to contrast than absolute luminance; therefore, we can perceive the world similarly regardless of the considerable changes in illumination conditions. Many algorithms for accomplishing contrast enhancement have been developed and applied to problems in image processing.

2. If the contrast of an image is highly concentrated on a specific range, e.g. an image is very dark; the information may be lost in those areas which are excessively and uniformly concentrated. The problem is to optimize the contrast of an image in order to represent all the information in the input image.

Arithmetic and logic operations

Image arithmetic applies one of the standard arithmetic operations or a logical operator to two or more images. The operators are applied in a pixel-by-pixel way, i.e. the value of a pixel in the output image depends only on the values of the corresponding pixels in the input images. Hence, the images must be of the same size. Although image arithmetic is the simplest form of image processing, there is a wide range of applications. A main advantage of arithmetic operators is that the process is very simple and therefore fast.

Logical operators are often used to combine two (mostly binary) images. In the case of integer images, the logical operator is normally applied in a bitwise way.

Spatial domain filtering

1. Filtering is a technique for modifying or enhancing an image. Spatial domain operation or filtering (the processed value for the current pixel depends on both itself and surrounding pixels). Hence Filtering is a neighborhood operation, in which the value of any given pixel in the output image is determined by applying some algorithm to the values of the pixels in the neighborhood of the corresponding input pixel. A

pixel's neighborhood is some set of pixels, defined by their locations relative to that pixel.

2. Spatial filtering is a form of finite impulse response (FIR) filtering. The filter is actually a mask of weights arranged in a rectangular pattern. The process is one of sliding the mask along the image and performing a multiply and accumulate operation on the pixels covered by the mask.

Image processing using python:

Images define the world, each image has its own story, it contains a lot of crucial information that can be useful in many ways. This information can be obtained with the help of the technique known as **Image Processing**.

It is the core part of computer vision which plays a crucial role in many real-world examples like robotics, self-driving cars, and object detection. Image processing allows us to transform and manipulate thousands of images at a time and extract useful insights from them. It has a wide range of applications in almost every field.

Python is one of the widely used programming languages for this purpose. Its amazing libraries and tools help in achieving the task of image processing very efficiently.

Through this article, you will learn about classical algorithms, techniques, and tools to process the image and get the desired output.

Let's get into it!

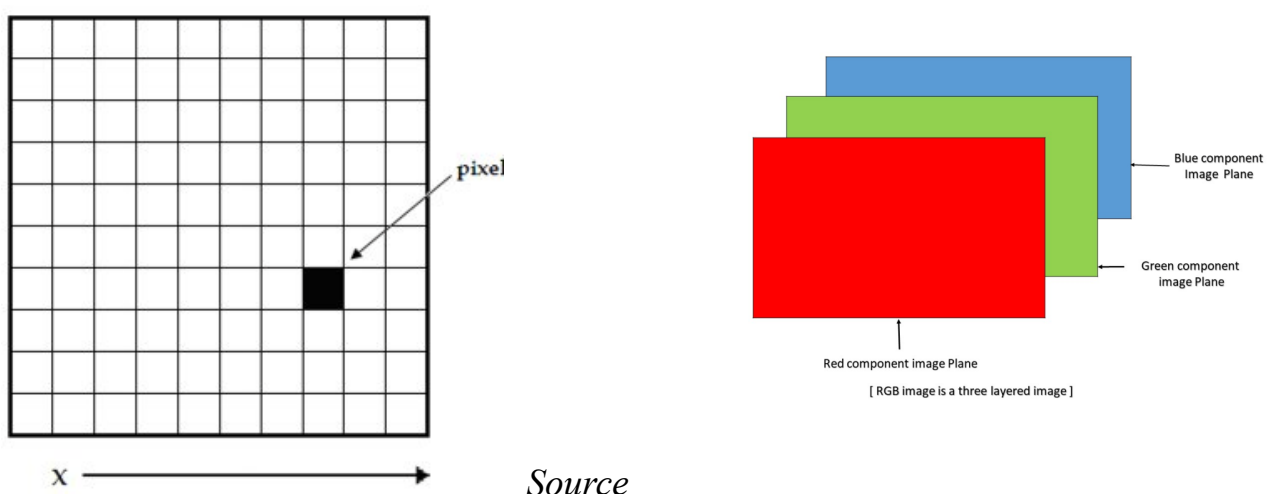
What is image processing?

As the name says, image processing means processing the image and this may include many different techniques until we reach our goal.

The final output can be either in the form of an image or a corresponding feature of that image. This can be used for further analysis and decision making.

But what is an image?

An image can be represented as a 2D function $F(x,y)$ where x and y are spatial coordinates. The amplitude of F at a particular value of x,y is known as the intensity of an image at that point. If x,y , and the amplitude value is finite then we call it a digital image. It is an array of pixels arranged in columns and rows. Pixels are the elements of an image that contain information about intensity and color. An image can also be represented in 3D where x,y , and z become spatial coordinates. Pixels are arranged in the form of a matrix. This is known as an **RGB image**.



There are various types of images:

- RGB image: It contains three layers of 2D image, these layers are Red, Green, and Blue channels.
- Grayscale image: These images contain shades of black and white and contain only a single channel.

Classic image processing algorithms

1. Morphological Image Processing

Morphological image processing tries to remove the imperfections from the binary images because binary regions produced by simple thresholding can be distorted by noise. It also helps in smoothing the image using opening and closing operations.

Morphological operations can be extended to grayscale images. It consists of non-linear operations related to the structure of features of an image. It depends on the related ordering of pixels but on their numerical values. This technique analyzes an image using a small template known as **structuring element** which is placed on different possible locations in the image and is compared with the corresponding neighbourhood pixels. A structuring element is a small matrix with 0 and 1 values.

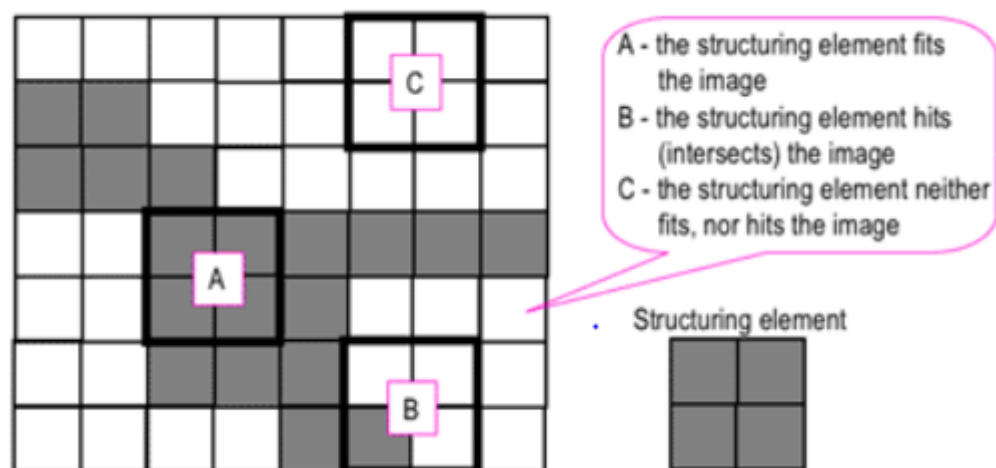
Let's see the two fundamental operations of morphological image processing, **Dilation and Erosion**:

- **dilation** operation adds pixels to the boundaries of the object in an image
- **erosion** operation removes the pixels from the object boundaries.

The number of pixels removed or added to the original image depends on the size of the structuring element.

At this point you may be thinking “what is a structuring element?” Let me explain:

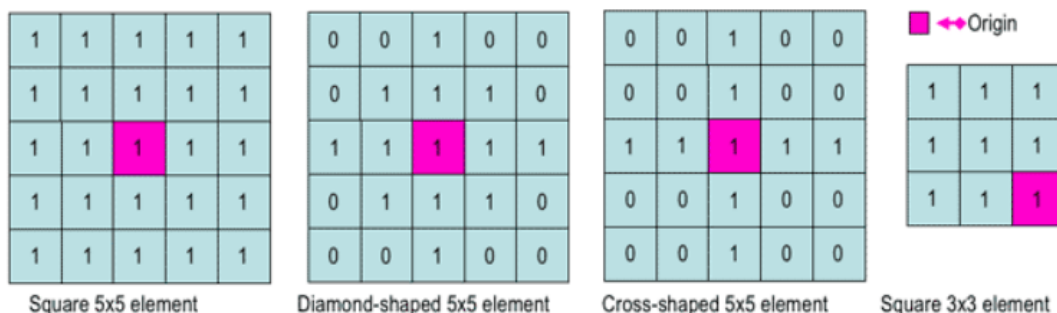
Structuring element is a matrix consisting of only 0’s and 1’s that can have any arbitrary shape and size. It is positioned at all possible locations in the image and it is compared with the corresponding neighbourhood of pixels.



Source :-

The square structuring element ‘A’ fits in the object we want to select, the ‘B’ intersects the object and ‘C’ is out of the object.

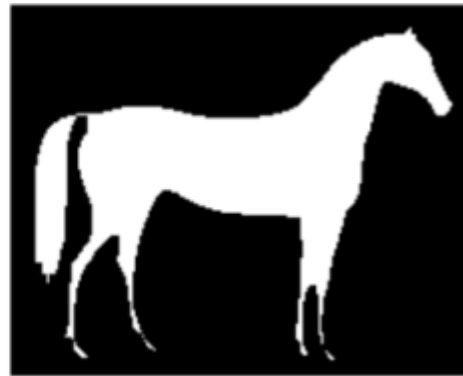
The zero-one pattern defines the configuration of the structuring element. It’s according to the shape of the object we want to select. The center of the structuring element identifies the pixel being processed.



Source



Dilation | [Source](#)



Erosion | [Source](#)

2. Gaussian Image Processing

Gaussian blur which is also known as gaussian smoothing, is the result of blurring an **image** by a **Gaussian** function.

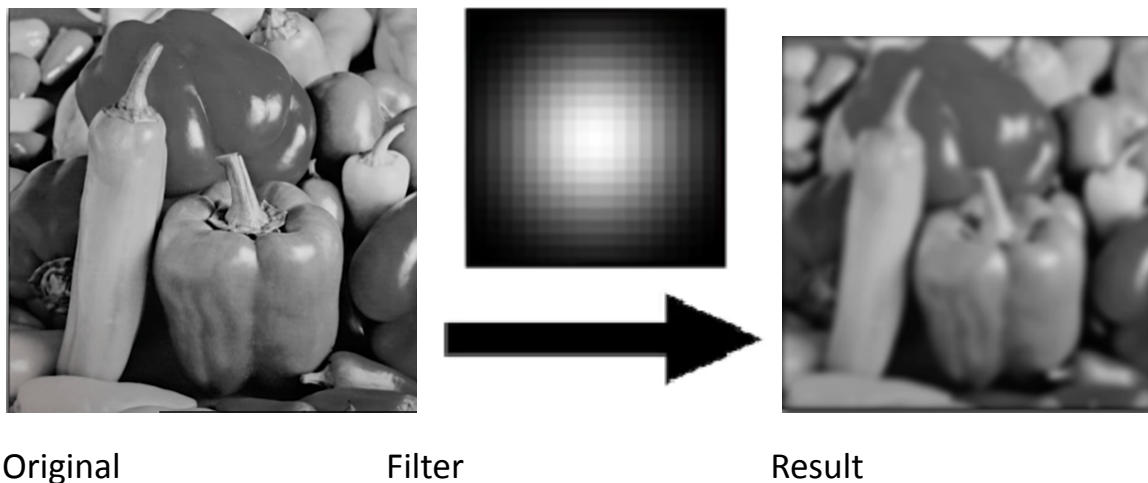
It is **used to reduce image noise and reduce details**. The visual effect of this blurring technique is similar to looking at an image through the translucent screen. It is sometimes used in computer vision for image enhancement at different scales or as a data augmentation technique in deep learning.

The basic gaussian function looks like:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$

In practice, it is best to take advantage of the Gaussian blur's separable property by dividing the process into two passes. In the first pass, a one-dimensional kernel is used to blur the image in only the horizontal or vertical direction. In the second pass, the same one-dimensional kernel is used to blur in the remaining direction. The resulting effect is the same as convolving with a two-dimensional kernel in a single pass. Let's see an example to understand what gaussian filters do to an image.

If we have a filter which is normally distributed, and when its applied to an image, the results look like this:



You can see that some of the edges have little less detail. The filter is giving more weight to the pixels at the center than the pixels away from the center. Gaussian filters are low-pass filters i.e. weakens the high frequencies. It is commonly used in edge detection.

3. Fourier Transform in image processing

Fourier transform breaks down an image into sine and cosine components.

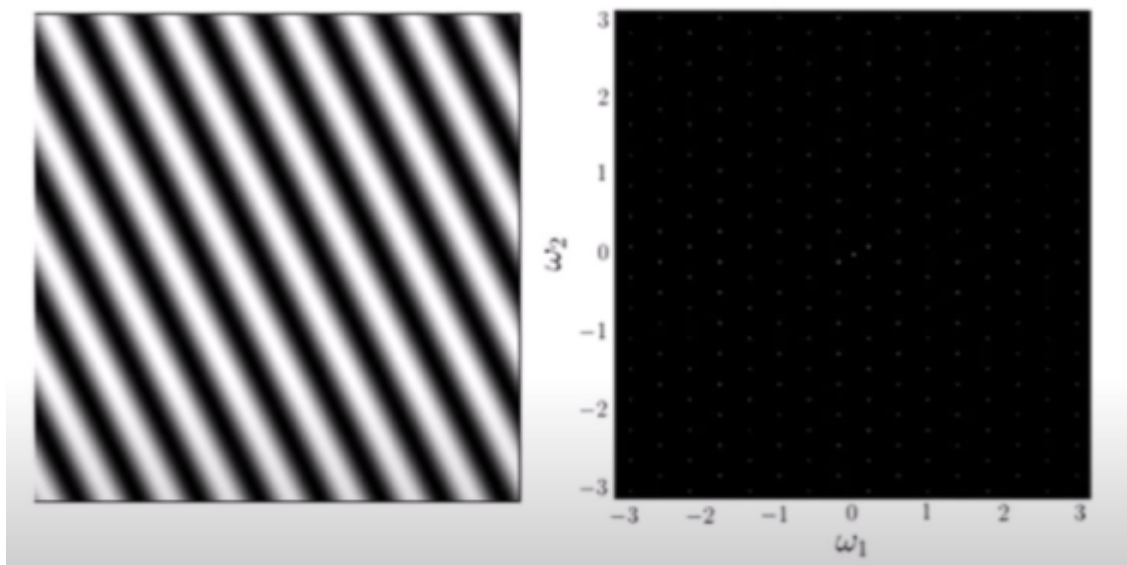
It has multiple applications like image reconstruction, image compression, or image filtering.

Since we are talking about images, we will take discrete fourier transform into consideration.

Let's consider a sinusoid, it comprises of three things:

- Magnitude – related to contrast
- Spatial frequency – related to brightness
- Phase – related to color information

The image in the frequency domain looks like this:



Source

The formula for 2D discrete fourier transform is:

$$F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(\frac{ux}{M} + \frac{vy}{N})}$$

In the above formula, $f(x, y)$ denotes the image.

The inverse fourier transform converts the transform back to image. The formula for 2D inverse discrete fourier transform is:

$$f(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(\frac{ux}{M} + \frac{vy}{N})}$$

4. Edge Detection in image processing

Edge detection is an image processing technique for finding the boundaries of objects within images. It works by detecting discontinuities in brightness.

This could be very beneficial in extracting useful information from the image because most of the shape information is enclosed in the edges. Classic edge detection methods work by detecting discontinuities in the brightness.

It can rapidly react if some noise is detected in the image while detecting the variations of grey levels. Edges are defined as the local maxima of the gradient.

The most common edge detection algorithm is **sobel edge detection algorithm**. Sobel detection operator is made up of 3*3 convolutional kernels. A simple kernel G_x and a 90 degree rotated kernel G_y . Separate measurements are made by applying both the kernel separately to the image.

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * Image\ matrix$$

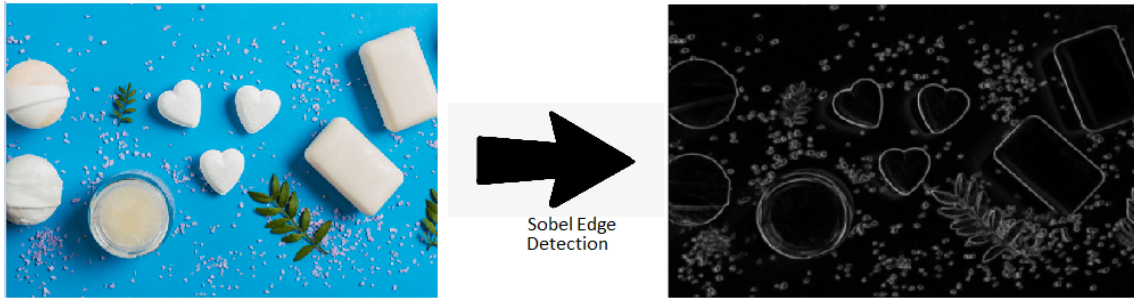
And,

$$G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * Image\ matrix$$

* denotes the 2D signal processing convolution operation.

Resulting gradient can be calculated as:

$$G = \text{sqrt}(Gx^2 + Gy^2)$$



Source

5. Wavelet Image Processing

We saw a Fourier transform but it is only limited to the frequency. Wavelets take both time and frequency into the consideration. This transform is apt for non-stationary signals.

We know that edges are one of the important parts of the image, while applying the traditional filters it's been noticed that noise gets removed but image gets blurry. The wavelet transform is designed in such a way that we get good frequency resolution for low frequency components. Below is the 2D wavelet transform example:



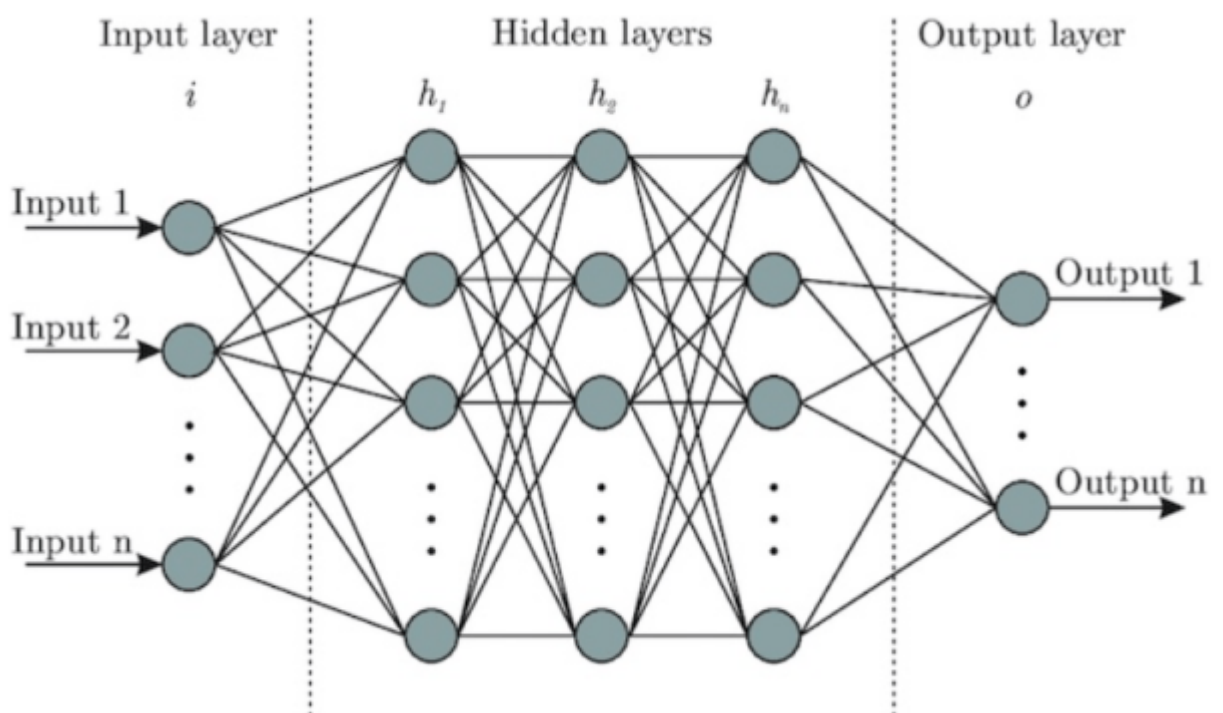
Source

Image processing using Neural Networks

Neural Networks are multi-layered networks consisting of neurons or nodes. These neurons are the core processing units of the neural network. They are designed to act like human brains. They take in data, train themselves to recognize the patterns in the data and then predict the output.

A basic neural network has three layers:

1. Input layer
2. Hidden layer
3. Output layer



Basic neural network :-

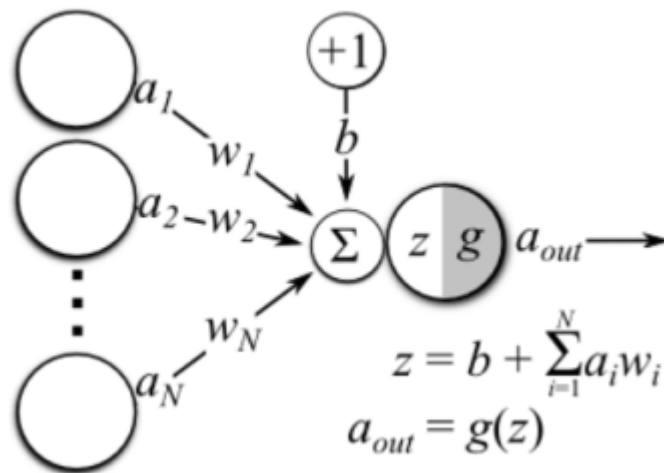
The input layers receive the input, the output layer predicts the output and the hidden layers do most of the calculations. The number of hidden layers can be

modified according to the requirements. There should be atleast one hidden layer in a neural network.

The basic working of the neural network is as follows:

1. Let's consider an image, each pixel is fed as input to each neuron of the first layer, neurons of one layer are connected to neurons of the next layer through channels.
2. Each of these channels is assigned a numerical value known as weight.
3. The inputs are multiplied by the corresponding weights and this weighted sum is then fed as input to the hidden layers.
4. The output from the hidden layers is passed through an activation function which will determine whether the particular neuron will be activated or not.
5. The activated neurons transmits data to the next hidden layers. In this manner, data is propagated through the network, this is known as Forward Propagation.
6. In the output layer, the neuron with the highest value predicts the output. These outputs are the probability values.
7. The predicted output is compared with the actual output to obtain the error. This information is then transferred back through the network, the process is known as Backpropagation.
8. Based on this information, the weights are adjusted. This cycle of forward and backward propagation is done several times on multiple inputs until the network predicts the output correctly in most of the cases.
9. This ends the training process of the neural network. The time taken to train the neural network may get high in some cases.

In the below image, **$\mathbf{a_i}$'s** is the set of inputs, **$\mathbf{w_i}$'s** are the weights, **$\mathbf{z}$** is the output and **$\mathbf{g}$** is any activation function.



Operations in a single

neuron | [Source](#)

Here are some guidelines to prepare data for image processing.

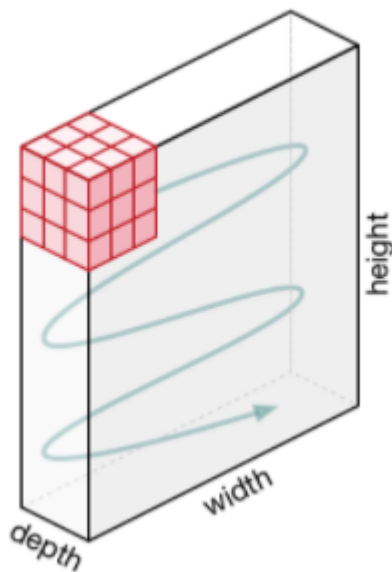
- More data needs to be fed to the model to get the better results.
- Image dataset should be of high quality to get more clear information, but to process them you may require deeper neural networks.
- In many cases RGB images are converted to grayscale before feeding them into a neural network.

Types of Neural Network

Convolutional Neural Network :-

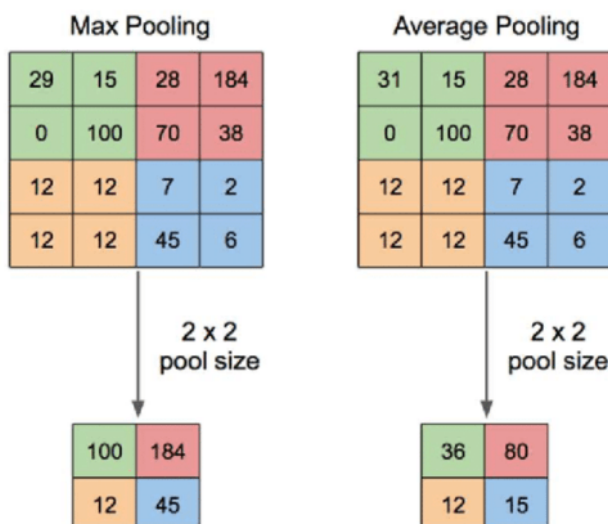
A convolutional neural network, ConvNets in short has three layers:

- **Convolutional Layer (CONV):** They are the core building block of CNN, it is responsible for performing convolution operation. The element involved in carrying out the convolution operation in this layer is called the **Kernel/Filter (matrix)**. The kernel makes horizontal and vertical shifts based on the **stride rate** until the full image is traversed.



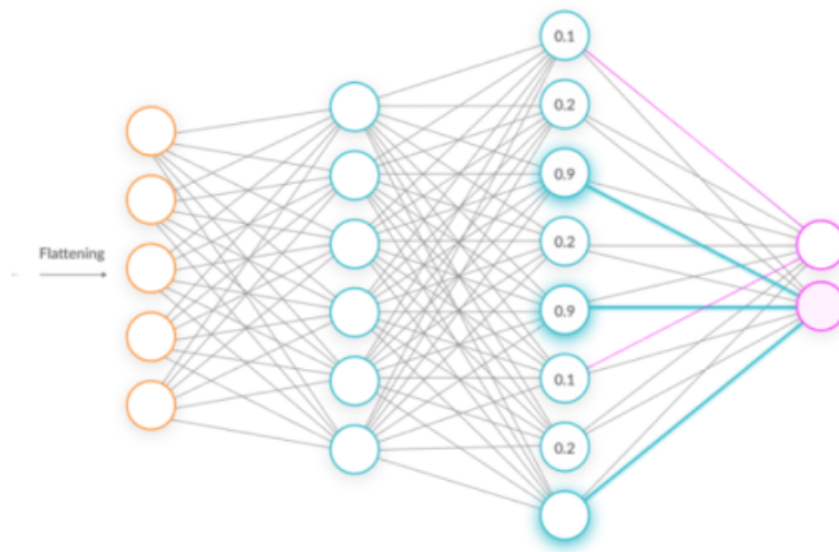
Movement of the kernel | [Source](#)

- Pooling Layer (POOL):** This layer is responsible for dimensionality reduction. It helps to decrease the computational power required to process the data. There are two types of Pooling: Max Pooling and Average Pooling. Max pooling returns the maximum value from the area covered by the kernel on the image. Average pooling returns the average of all the values in the part of the image covered by the kernel.



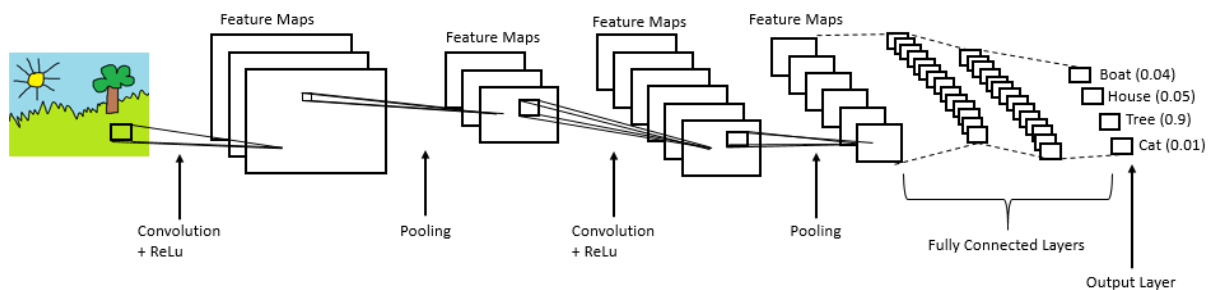
Pooling operation | [Source](#)

- Fully Connected Layer (FC):** The **fully connected layer (FC)** operates on a flattened input where each input is connected to all neurons. If present, **FC layers** are usually found towards the end of **CNN** architectures.



Fully connected layers

CNN is mainly used in extracting features from the image with help of its layers. CNNs are widely used in image classification where each input image is passed through the series of layers to get a probabilistic value between 0 and 1.



Source

Generative Adversarial Networks

Generative models use an unsupervised learning approach (there are images but there are no labels provided).

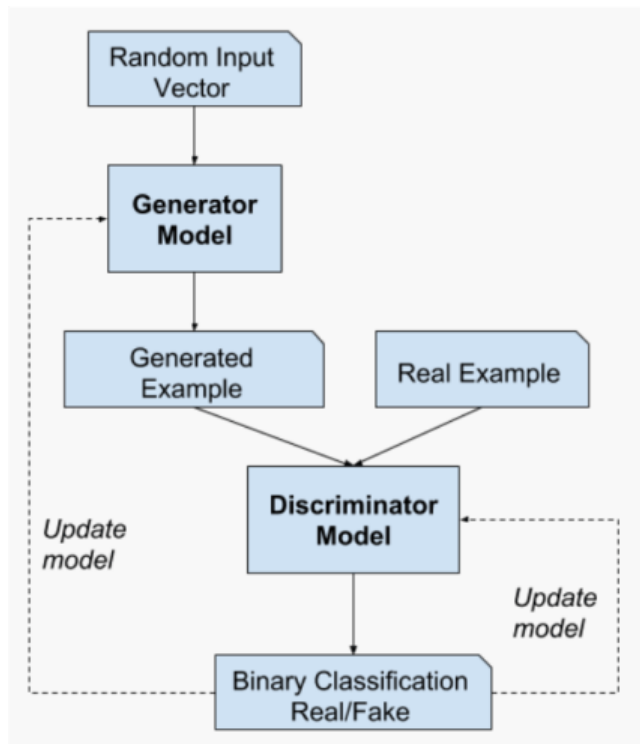
GANs are composed of two models **Generator** and **Discriminator**. *Generator* learns to make fake images that look realistic so as to fool the discriminator and *Discriminator* learns to distinguish fake from real images (it tries not to get fooled).

Generator is not allowed to see the real images, so it may produce poor results in the starting phase while the discriminator is allowed to look at real images but they are jumbled with the fake ones produced by the generator which it has to classify as real or fake.

Some noise is fed as input to the generator so that it's able to produce different examples every single time and not the same type image. Based on the scores predicted by the discriminator, the generator tries to improve its results, after a certain point of time, the generator will be able to produce images that will be harder to distinguish, at that point of time, the user gets satisfied with its results. Discriminator also improves itself as it gets more and more realistic images at each round from the generator.

Popular types of GANs are Deep Convolutional GANs(DCGANs), Conditional GANs(cGANs), StyleGANs, CycleGAN, DiscoGAN, GauGAN and so on.

GANs are great for image generation and manipulation. Some applications of GANs include : Face Aging, Photo Blending, Super Resolution, Photo Inpainting, Clothing Translation.



Source

Image processing tools

1. OpenCV

It stands for Open Source Computer Vision Library. This library consists of around 2000+ optimised algorithms that are useful for computer vision and machine learning. There are several ways you can use opencv in image processing, a few are listed below:

- Converting images from one color space to another i.e. like between BGR and HSV, BGR and gray etc.
- Performing thresholding on images, like, simple thresholding, adaptive

thresholding etc.

- Smoothing of images, like, applying custom filters to images and blurring of images.
- Performing morphological operations on images.
- Building image pyramids.
- Extracting foreground from images using GrabCut algorithm.
- Image segmentation using watershed algorithm.

Refer to [this link](#) for more details.

2. Scikit-image

It is an open-source library used for image preprocessing. It makes use of machine learning with built-in functions and can perform complex operations on images with just a few functions.

It works with numpy arrays and is a fairly simple library even for those who are new to python. Some operations that can be done using scikit image are :

- To implement thresholding operations use **try_all_threshold()** method on the image. It will use seven global thresholding algorithms. This is in the **filters** module.
- To implement edge detection use **sobel()** method in the **filters** module. This method requires a 2D grayscale image as an input, so we need to convert the image to grayscale.
- To implement gaussian smoothing use **gaussian()** method in the **filters** module.
- To apply histogram equalization, use **exposure** module, to apply normal histogram equalization to the original image, use **equalize_hist()** method and to apply adaptive equalization, use **equalize_adapthist()** method.
- To rotate the image use **rotate()** function under the **transform** module.
- To rescale the image use **rescale()** function from the **transform** module.

- To apply morphological operations use **binary_erosion()** and **binary_dilation()** function under the **morphology** module.

3. PIL/pillow

PIL stands for Python Image Library and **Pillow** is the friendly PIL fork by Alex Clark and Contributors. It's one of the powerful libraries. It supports a wide range of image formats like PPM, JPEG, TIFF, GIF, PNG, and BMP.

It can help you perform several operations on images like rotating, resizing, cropping, grayscaling etc. Let's go through some of those operations

To carry out manipulation operations there is a module in this library called **Image**.

- To load an image use the **open()** method.
- To display an image use **show()** method.
- To know the file format use **format** attribute
- To know the size of the image use **size** attribute
- To know about the pixel format use **mode** attribute.
- To save the image file after desired processing, use **save()** method. Pillow saves the image file in *png* format.
- To resize the image use **resize()** method that takes two arguments as width and height.
- To crop the image, use **crop()** method that takes one argument as a box tuple that defines position and size of the cropped region.
- To rotate the image use **rotate()** method that takes one argument as an integer or float number representing the degree of rotation.
- To flip the image use **transform()** method that take one argument among the following: `Image.FLIP_LEFT_RIGHT`, `Image.FLIP_TOP_BOTTOM`,

Image.ROTATE_90, Image.ROTATE_180, Image.ROTATE_270.

4. NumPy

With this library you can also perform simple image techniques, such as flipping images, extracting features, and analyzing them.

Images can be represented by numpy multi-dimensional arrays and so their type is **NdArrays**. A color image is a numpy array with 3 dimensions. By slicing the multi-dimensional array the RGB channels can be separated.

Below are some of the operations that can be performed using NumPy on the image (image is loaded in a variable named **test_img** using imread).

- To flip the image in a vertical direction, use **np.flipud(test_img)**.
- To flip the image in a horizontal direction, use **np.fliplr(test_img)**.
- To reverse the image, use **test_img[::-1]** (the image after storing it as the numpy array is named as <img_name>).
- To add filter to the image you can do this:

Example: **np.where(test_img > 150, 255, 0)**, this says that in this picture if you find anything with 150, then replace it with 255, else 0.

- You can also display the RGB channels separately. It can be done using this code snippet:

To obtain a red channel, do **test_img[:, :, 0]**, to obtain a green channel, do **test_img[:, :, 1]** and to obtain a blue channel, do **test_img[:, :, 2]**.

5. Mahotas

It is a computer vision and image processing library and has more than 100 functions. Many of its algorithms are implemented in C++. Mahotas is an independent module in itself i.e. it has minimal dependencies.

Currently, it depends only on C++ compilers for numerical computations, there is no need for NumPy module, the compiler does all its work.

Here are names of some of the remarkable algorithms available in Mahotas:

- Watershed (<https://mahotas.readthedocs.io/en/latest/distance.html>)
- Morphological Operations
(<https://mahotas.readthedocs.io/en/latest/morphology.html>)
- Hit & miss, thinning.
(<https://mahotas.readthedocs.io/en/latest/api.html#mahotas.hitmiss>)
- Colorspace Conversions (<https://mahotas.readthedocs.io/en/latest/color.html>)
- Speeded-Up Robust Features (SURF), a form of local features.
(<https://mahotas.readthedocs.io/en/latest/surf.html>)
- Thresholding. (<https://mahotas.readthedocs.io/en/latest/thresholding.html>)
- Convolution. (<https://mahotas.readthedocs.io/en/latest/api.html>)
- Spline interpolation (<https://mahotas.readthedocs.io/en/latest/api.html>)
- SLIC superpixels.
(<https://www.pyimagesearch.com/2014/07/28/a-slic-superpixel-tutorial-using-python/>)

Let's look at some of the operations that could be done using Mahotas:

- To read an image use **imread()** method.
- To calculate the mean of the image use the **mean()** method.
- Eccentricity of an image measures the shortest length of the paths from a given vertex *v* to reach any other vertex *w* of a connected graph. To find the eccentricity of an image, use the **eccentricity()** method under the **features** module.

- For dilation and erosion on the image use, **dilate()** and **erode()** method under **morph** module.
- To find the local maxima of the image use **locmax()** method.

Algorithm:

1) Greedy search :-

A greedy algorithm is a mathematical process that looks for simple, easy-to-implement solutions to complex, multi-step problems by deciding which next step will provide the most obvious benefit.

Such algorithms are called greedy because while the optimal solution to each smaller instance will provide an immediate output, the algorithm doesn't consider the larger problem as a whole. Once a decision has been made, it is never reconsidered. Greedy algorithms work by recursively constructing a set of objects from the smallest possible constituent parts. Recursion is an approach to problem solving in which the solution to a particular problem depends on solutions to smaller instances of the same problem.

Advantage: The advantage to using a greedy algorithm is that solutions to smaller instances of the problem can be straightforward and easy to understand.

Disadvantage: The disadvantage is that it is entirely possible that the most optimal short-term solutions may lead to the worst possible long-term outcome.

2) Beam Search :-

Beam search is an algorithm used in many NLP and speech recognition models as a final decision making layer to choose the best output given target variables like maximum probability or next output character.

A beam search takes three components as its input: a problem to be solved, a set of heuristic rules for pruning, and a memory with a limited available capacity. The problem is the problem to be solved, usually represented as a graph, and contains a set of nodes in which one or more of the nodes represents a goal. The set of heuristic rules are rules specific to the problem domain and prune unfavourable nodes from the memory in respect to the problem domain.

Advantage: Beam search has the advantage of potentially reducing the computation, and hence the time, of a search and the memory consumption of the search is far less than its underlying search methods

Disadvantage: The main disadvantages of a beam search are that the search may not result in an optimal goal and may not even reach a goal at all.

Python Code :-

```
import tkinter as tk

from tkinter import filedialog

from tkinter import *

from PIL import ImageTk, Image

import numpy as np

import cv2

from gtts import gTTS

import os

#load the trained model to classify sign

from tensorflow.keras.models import load_model
```

```

from tensorflow.keras.preprocessing.sequence import pad_sequences

from tensorflow.keras.models import Model

from tensorflow.keras.applications.inception_v3 import InceptionV3,
preprocess_input

from pickle import dump, load

from tensorflow.keras.preprocessing.image import load_img, img_to_array


base_model = InceptionV3(weights =
'inception_v3_weights_tf_dim_ordering_tf_kernels.h5')

vgg_model = Model(base_model.input, base_model.layers[-2].output)


def preprocess_img(img_path):

    #inception v3 expects img in 299*299

    img = load_img(img_path, target_size = (299, 299))

    x = img_to_array(img)

    # Add one more dimension

    x = np.expand_dims(x, axis = 0)

    x = preprocess_input(x)

    return x


def encode(image):

    image = preprocess_img(image)

    vec = vgg_model.predict(image)

```

```
vec = np.reshape(vec, (vec.shape[1]))
```

```
return vec
```

```
pickle_in = open("wordtoix.pkl", "rb")
```

```
wordtoix = load(pickle_in)
```

```
pickle_in = open("ixtoward.pkl", "rb")
```

```
ixtoward = load(pickle_in)
```

```
max_length = 74
```

```
def greedy_search(pic):
```

```
    start = 'startseq'
```

```
    for i in range(max_length):
```

```
        seq = [wordtoix[word] for word in start.split() if word in wordtoix]
```

```
        seq = pad_sequences([seq], maxlen = max_length)
```

```
        yhat = model.predict([pic, seq])
```

```
        yhat = np.argmax(yhat)
```

```
        word = ixtoward[yhat]
```

```
        start += ' ' + word
```

```
        if word == 'endseq':
```

```
            break
```

```
    final = start.split()
```

```
final = final[1:-1]
```

```
final = ''.join(final)
```

```
return final
```

```
def beam_search(image, beam_index = 3):
```

```
    start = [wordtoix["startseq"]]
```

```
    # start_word[0][0] = index of the starting word
```

```
    # start_word[0][1] = probability of the word predicted
```

```
    start_word = [[start, 0.0]]
```

```
    while len(start_word[0][0]) < max_length:
```

```
        temp = []
```

```
        for s in start_word:
```

```
            par_caps = pad_sequences([s[0]], maxlen=max_length)
```

```
            e = image
```

```
            preds = model.predict([e, np.array(par_caps)])
```

```
            # Getting the top <beam_index>(n) predictions
```

```
            word_preds = np.argsort(preds[0])[-beam_index:]
```

```
            # creating a new list so as to put them via the model again
```

```
for w in word_preds:

    next_cap, prob = s[0][:], s[1]

    next_cap.append(w)

    prob += preds[0][w]

    temp.append([next_cap, prob])
```

```
start_word = temp
```

```
# Sorting according to the probabilities
```

```
start_word = sorted(start_word, reverse=False, key=lambda l: l[1])
```

```
# Getting the top words
```

```
start_word = start_word[-beam_index:]
```

```
start_word = start_word[-1][0]
```

```
intermediate_caption = [ixtoword[i] for i in start_word]
```

```
final_caption = []
```

```
for i in intermediate_caption:
```

```
    if i != 'endseq':
```

```
        final_caption.append(i)
```

```
    else:
```

```
        break
```

```
final_caption = ''.join(final_caption[1:])
```

```
return final_caption
```

```
model = load_model('new-model-1.h5')
```

```
#initialise GUI
```

```
top=tk.Tk()
```

```
top.geometry('800x600')
```

```
top.title('Caption Generator')
```

```
top.configure(background='#CDCDCD')
```

```
image = PhotoImage(file="logo.png", width=860, height=180)
```

```
#We have to put that image in that label
```

```
labelImg = Label(image=image)
```

```
#now we have To pack that Image
```

```
labelImg.pack()
```

```
label2=Label(top,background='#CDCDCD', font=('arial',15))
```

```
label1=Label(top,background='#CDCDCD', font=('arial',15))
```

```
label=Label(top,background='#CDCDCD', font=('arial',15))
```

```
sign_image = Label(top)
```

```

def classify(file_path):

    global label_packed

    enc = encode(file_path)

    image = enc.reshape(1, 2048)

    language = 'en'

    pred = greedy_search(image)

    print(pred)

    label.configure(foreground='#000', text= 'Greedy: ' + pred)

    label.pack(side=BOTTOM,expand=True)

    beam_3 = beam_search(image)

    print(beam_3)

    label1.configure(foreground='#011638', text = 'Beam 1: ' + beam_3)

    label1.pack(side = BOTTOM, expand = True)

    beam_5 = beam_search(image, 5)

    print(beam_5)

    output=gTTS(text=beam_5,lang=language,slow=False)

    output.save("output.mp3")

    os.system("start output.mp3")

    label2.configure(foreground='#228B22', text = 'Final: ' + beam_5)

    label2.pack(side = BOTTOM, expand = True)


def show_classify_button(file_path):

```

```
classify_b=Button(top,text="Generate Audio Caption",command=lambda:
classify(file_path),padx=10,pady=5)
```

```
classify_b.configure(background='#364156',
foreground='white',font=('arial',10,'bold'))
```

```
classify_b.place(relx=0.79,rely=0.46)
```

```
def upload_image():
```

```
    try:
```

```
        file_path=filedialog.askopenfilename()
```

```
        uploaded=Image.open(file_path)
```

```
        uploaded.thumbnail(((top.winfo_width()/2.25),(top.winfo_height()/2.25)))
```

```
        im=ImageTk.PhotoImage(uploaded)
```

```
        sign_image.configure(image=im)
```

```
        sign_image.image=im
```

```
        label.configure(text="")
```

```
        label1.configure(text="")
```

```
        label2.configure(text="")
```

```
        show_classify_button(file_path)
```

```
    except:
```

```
        pass
```

```
upload=Button(top,text="Upload an
image",command=upload_image,padx=10,pady=5)
```

```
upload.configure(background='#364156',
foreground='white',font=('arial',10,'bold'))
```

```
upload.pack(side=BOTTOM,pady=50)
```

```
sign_image.pack(side=BOTTOM,expand=True)
```

```
#label2.pack(side = BOTTOM, expand = True)
```

```
heading = Label(top, text="Caption Generator Application",pady=20,  
font=('arial',22,'bold'))
```

```
heading.configure(background='#CDCDCD',foreground='#FF6347')
```

```
heading.pack()
```

```
top.mainloop()
```

Result screenshots:

1:

Caption Generator


SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

Caption Generator Application

Final: group of people are working at computers

Beam 1: group of people are working at computers

Greedy: group of people are sitting at table in front of computers

[Generate Audio Caption](#)



[Upload an image](#)

2:

Beam_5: group of colorfully clothed people lie on the ground in front of an ornate building

Beam_3: group of people are sitting in front of building

Greedy: two children are playing with toy guns



[Generate](#)

[Upload an image](#)

3:

Beam_5: two men in hard hats are working on scaffolding

Beam_3: two men in hard hats are working on scaffolding

Greedy: man in blue shirt is standing on top of large rock



Generate

Upload an image

4:

Beam_5: group of three men are playing guitars on stage

Beam_3: two men are playing guitars on stage

Greedy: man in black shirt is playing guitar



Generate

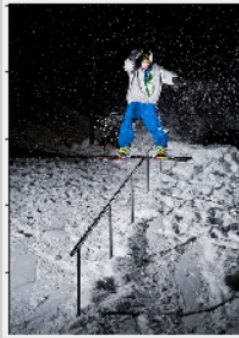
Upload an image

5:

Beam_5: snowboarder is jumping over stair railing at night

Beam_3: man in red shirt is snowboarding down the stairs

Greedy: man is skateboarding on ramp



Generate

Upload an image

Conclusion:

We successfully completed what we mentioned in the project proposal, but used a smaller dataset (Flickr8k) due to limited computational power. There can be potential improvements if given more time.

First of all, we directly used pre-trained CNN network as part of our pipeline without fine-tuning, so the network does not adapt to this specific training dataset. Thus, by experimenting with different CNN pre-trained networks and enabling fine-tuning, we expect to achieve a slightly better predicted caption.

Another potential improvement is by training on a combination of Flickr8k, Flickr30k, and MSCOCO. In general, the more diverse training dataset, the more accurate the output will be.

SCOPE:

The scope of image-captioning is very vast in the future as the users are increasing day by day on social media and most of them would post photos. So this project will help them to a greater extent.

References:

- [1] Pranay Mathur, Aman Gill, Aayush Yadav, Anurag Mishra and Nand Kumar Bansode (2017) International Conference on Computational Intelligence in Data Science (ICCIDS) : Camera2Caption: A Real-Time Image Caption Generator
- [2] Simao Herdade, Armin Kappeler, Kofi Boakye, Joao Soares (june 2019) : Image Captioning: Transforming Objects into Words.
- [3] R. Subash November 2019 Journal of Physics Conference Series 1362:012096 : Automatic Image Captioning Using Convolution Neural Networks and LSTM.
- [4] B. Krishnakumar, K. Kousalya, S. Gokul, R. Karthikeyan, D. Kaviyarasu (IJAST) Vol.29 NO.3s(2020) : Image Caption Generation Using Deep Learning.
- [5] Seung-Ho Han, Ho-Jin Choi (2020) IEEE International Conference on Big Data and Smart Computing (BigComp) : Domain-Specific Image Caption Generator with Semantic Ontology