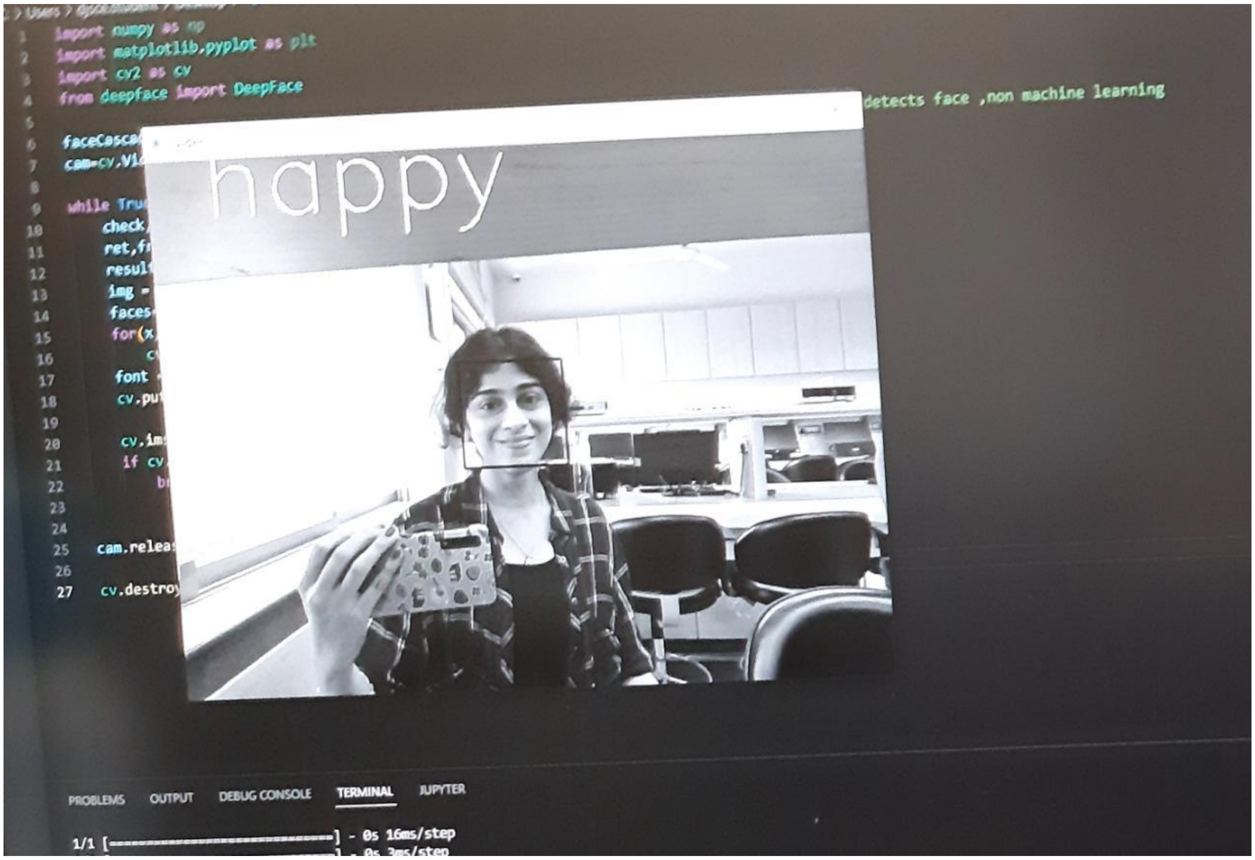
DJ SYNAPSE- ML TASK 3




**ARYA TELANG**

**SE-C COMPUTER ENGINEERING**

**SAPId:60004210201**

# CV: Facial Recognition & Emoji Generation

# Preprocessing Steps

**Image processing could be simple tasks like image resizing. In order to feed a dataset of images to a convolutional network, they must all be the same size. Other processing tasks can take place like geometric and color transformation or converting color to grayscale and many more**

## 1.Denoising

- In image processing, noise means any undesirable change in pixel value.
- Gaussian blur is the process of blurring an image using the gaussian function. It is widely used in graphics software to remove noise from the image and reduce detail.
- Gaussian filters have the properties of having no overshoot to a step function input while minimizing the rise and fall time. In terms of image processing, any sharp edges in images are smoothed while minimizing too much blurring.

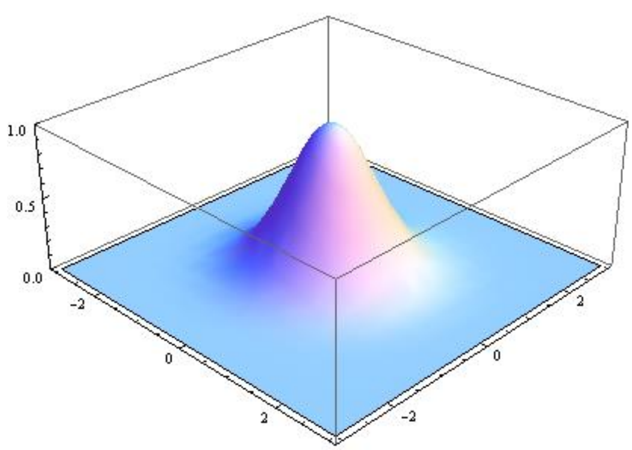
**Syntax:**

```
dst = cv2.GaussianBlur(src, ksize, sigmaX[, dst[, sigmaY[,
borderType=BORDER_DEFAULT]]] )
```

- src- input image
- dst-output image
- ksize=Gaussian kernel size

**Mathematical Equations used:**

$$G(x,y)=\frac{1}{2\pi\sigma 2}e^{-(x2+y2)/2\sigma 2}$$



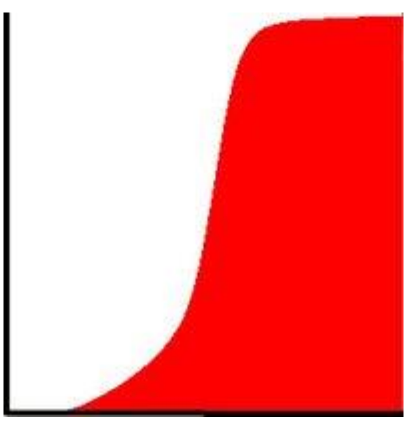

Gaussian Function

# 2.Contrast Enhancement

- If grey level image is too dark or too bright, this may be applied.
- It is a graphical representation of the intensity distribution of an image.
- It is a method that improves the contrast in an image, in order to stretch out the intensity range.
- To make it clearer, from the image above, you can see that the pixels seem clustered around the middle of the available range of intensities. What Histogram Equalization does is to *stretch out* this range.
- Equalization implies *mapping* one distribution (the given histogram) to another distribution (a wider and more uniform distribution of intensity values) so the intensity values are spreaded over the whole range.
- To accomplish the equalization effect, the remapping should be the *cumulative distribution function (cdf)* (more details, refer to *Learning OpenCV*). For the histogram $H(i)$, its *cumulative distribution* $H'(i)$ is:

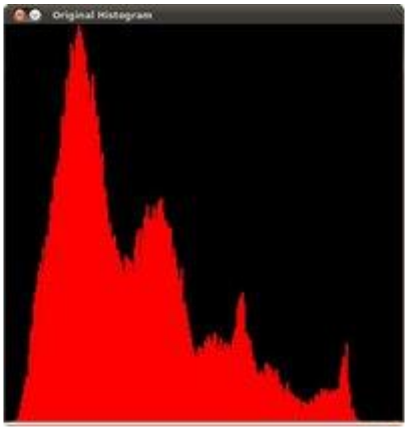
$$H'(i) = \sum_{0 \le j < i} H(j)$$

To use this as a remapping function, we have to normalize $H'(i)$ such that the maximum value is 255 ( or the maximum value for the intensity of the image ). From the example above, the cumulative function is:

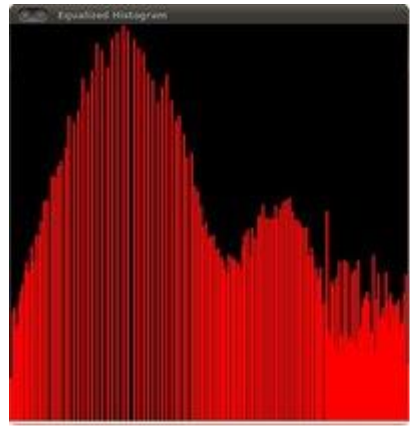


- Finally, we use a simple remapping procedure to obtain the intensity values of the equalized image:

$$\mathrm{equalized}(x,y) = H'(\mathrm{src}(x,y))$$



Initial Histogram: Pixels clustered around Centre



Normalized Histogram-Pixels spread out

# 3.Grey Scale

- <u>Greyscaling the image</u>: To change the gray-scale image into a colored one, use the syntax

<u>Syntax:</u>

```
gray_image =cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

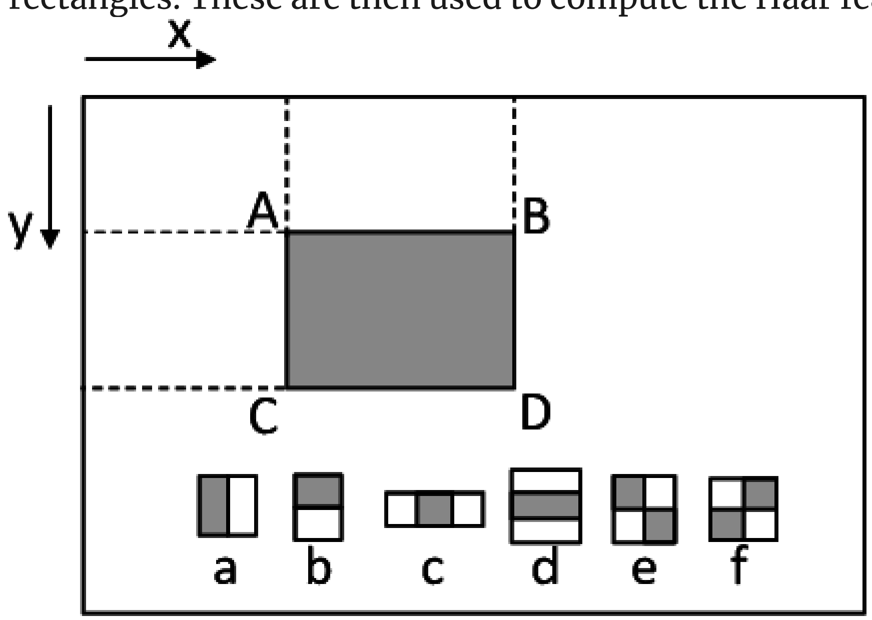
- Gray scale is simply **reducing code complexity**: from a 3D pixel value (R,G,B) to a 1D value. When you convert a RGB image into Gray scale you discard lots of information which are not required for processing. Incase of a RGB scale image, for each of the component, i.e. R,G,B, image holds different intensity labels. RGB image is represented by 3 channels. each of the channel generally consists of 8-bits.
- Increases speed-working on a single image and we process it in a triple channel such as RGB, it takes about 3 times longer than in Grayscale

# 4.Edge Detection Using Canny Algorithm

- Edges are representations of the shape of the object present in the image. Once edges are detected, we can use enhancement techniques to a particular region to extract more features.
- 
- Canny edge detection is based on:
- gradient of the image i.e. difference between two adjacent pixels
- hysteresis filtering: It selects the lines using those pixels which are different from adjacent ones.
- **Gaussian Filter:** Smooth the input image with a Gaussian filter to remove noise (using a discrete Gaussian kernel).
- **Calculate Intensity Gradients:** Identify the areas in the image with the strongest intensity gradients
- **Non-maximum Suppression:** Apply non-maximum suppression to thin out the edges. We want to remove unwanted pixels that might not be part of an edge.
- **Thresholding with Hysteresis:** Hysteresis or double thresholding involves:
    - Accepting pixels as edges if the intensity gradient value exceeds an upper threshold.
    - Rejecting pixels as edges if the intensity gradient value is below a lower threshold.
    - If a pixel is between the two thresholds, accept it only if it is adjacent to a pixel that is above the upper threshold.

# Image Classification Techniques

# 1.Haar cascade classifiers

- Haar Cascade classifiers are an effective way for object detection
- Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.
- Works with face detection. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it.
- The repository has the models stored in XML files, and can be read with the OpenCV methods. These include models for face detection, eye detection, upper body and lower body detection, license plate detection etc.
- Integral images essentially speed up the calculation of these Haar features. Instead of computing at every pixel, it instead creates sub-rectangles and creates array references for each of those sub-rectangles. These are then used to compute the Haar features.
- 

**Integral Image:**

Rectangle features can be computed very rapidly using an intermediate representation for the image which we call the integral image.[2] The integral image at location $x, y$ contains the sum of the pixels above and to the left of $x, y$, inclusive:

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y'),$$

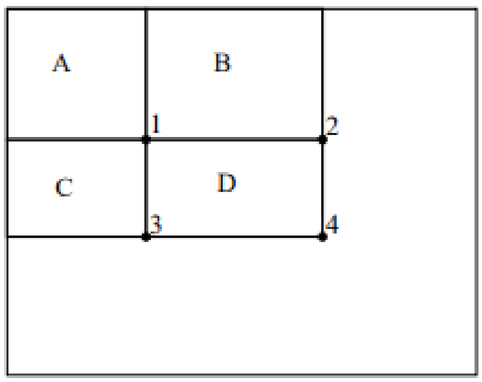



Figure 2: The sum of the pixels within rectangle $D$ can be computed with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle $A$. The value at location 2 is $A + B$, at location 3 is $A + C$, and at location 4 is $A + B + C + D$. The sum within $D$ can be computed as $4 + 1 - (2 + 3)$.

where $ii(x, y)$ is the integral image and $i(x, y)$ is the original image. Using the following pair of recurrences:

$$s(x, y) = s(x, y - 1) + i(x, y) \quad (1)$$

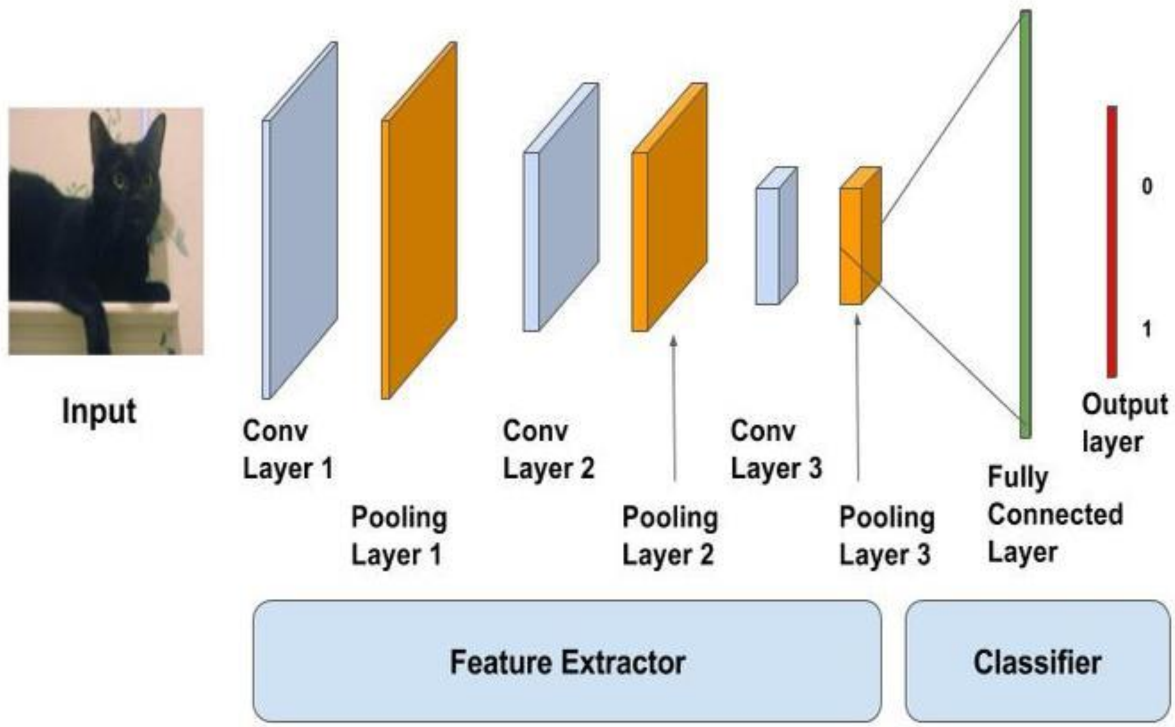
$$ii(x, y) = ii(x - 1, y) + s(x, y) \quad (2)$$

(where $s(x, y)$ is the cumulative row sum, $s(x, -1) = 0$, and $ii(-1, y) = 0$) the integral image can be computed in one pass over the original image.

Using the integral image any rectangular sum can be computed in four array references (see Figure 2). Clearly the difference between two rectangular sums can be computed in eight references. Since the two-rectangle features defined above involve adjacent rectangular sums they can be computed in six array references, eight in the case of the three-rectangle features, and nine for four-rectangle features.

# 2.Convolutional Neural Networks

- Convolutional Neural Networks come under the subdomain of Machine Learning which is Deep Learning. Algorithms under Deep Learning process information the same way the human brain does, but obviously on a very small scale, since our brain is too complex
- Image classification involves the extraction of features from the image to observe some patterns in the dataset. Using an ANN for the purpose of image classification would end up being very costly in terms of computation since the trainable parameters become extremely large.
- Convolution basically means a pointwise multiplication of two functions to produce a third function. Here one function is our image pixels matrix and another is our filter. We slide the filter over the image and get the dot product of the two matrices. The resulting matrix is called an "Activation Map" or "Feature Map".
- The Convolutional Neural Network takes a different approach, mimicking the way we perceive our environment with our eyes. When we see an image, we automatically divide it into many small sub-images and analyze them one by one. By assembling these sub-images, we process and interpret the image.
- The work happens in the so-called convolution layer. To do this, we define a filter that determines how large the partial images we are looking at should be, and a step length that decides how many pixels we continue between calculations, i.e. how close the partial images are to each other. By taking this step, we have greatly reduced the dimensionality of the image.
- The next step is the pooling layer. From a purely computational point of view, the same thing happens here as in the convolution layer, with the difference that we only take either the average or maximum value from the result, depending on the application. This preserves small features in a few pixels that are crucial for the task solution.
- Finally, there is a fully-connected layer, as we already know it from the normal neural networks. Now that we have greatly reduced the dimensions of the image, we can use the tightly meshed layers. Here, the individual sub-images are linked again in order to recognize the connections and to carry out the classification.




# CONCLUSION

- A dataset in computer vision is a curated set of digital photographs that developers use to test, train and evaluate the performance of their algorithms. The algorithm is said to learn from the examples contained in the dataset
- Pre-processing is required to clean image data for model input. Fully connected layers in convolutional neural networks required that all images are the same sized arrays.
- Image pre-processing may also decrease model training time and increase model inference speed. If input images are particularly large, reducing the size of these images will dramatically improve model training time without significantly reducing model performance.
- Image augmentation creates new training examples out of existing training data. It’s impossible to truly capture an image that accounts for every real world scenario a model may encompass. Adjusting existing training data to generalize to other situations allows the model to learn from a wider array of situations.

- This is particularly important when collected datasets may be small. A deep learning model will (over)fit to the examples shown in training, so creating variation in the input images enables generation of new, useful training examples.
- Coming to object classifiers:
- The CNN detector was able to detect a larger variety of faces. Even if one tilts their face, turn it partially away from the camera, or partially obscure it with ones' hands, it was still able to recognize it as a face. The OpenCV Haar-based classifier could only really recognize full front-facing faces.
- The emotion recognition network, trained in accordance with the Haar-based classifier, could only accurately recognize different emotions on full front-facing faces. Hence, even if the CNN detector allowed us to draw a bounding box around partially obscured faces, the program couldn't really recognize the emotion on the face.