x-ray image analysis using cnn

Dataset Link:https://www.kaggle.com/datasets/alifrahman/covid19-chest-xray-image-dataset

import necessary libraries

```
!pip install kaggle
```

```
Requirement already satisfied: kaggle in /usr/local/lib/python3.12/dist-packages (1.7.4.5)
Requirement already satisfied: bleach in /usr/local/lib/python3.12/dist-packages (from kaggle) (6.2.0)
Requirement already satisfied: certifi>=14.05.14 in /usr/local/lib/python3.12/dist-packages (from kaggle) (2025.8.3)
Requirement already satisfied: charset-normalizer in /usr/local/lib/python3.12/dist-packages (from kaggle) (3.4.3)
Requirement already satisfied: idna in /usr/local/lib/python3.12/dist-packages (from kaggle) (3.10)
Requirement already satisfied: protobuf in /usr/local/lib/python3.12/dist-packages (from kaggle) (5.29.5)
Requirement already satisfied: python-dateutil>=2.5.3 in /usr/local/lib/python3.12/dist-packages (from kaggle) (2.9.0.post0)
Requirement already satisfied: python-slugify in /usr/local/lib/python3.12/dist-packages (from kaggle) (8.0.4)
Requirement already satisfied: requests in /usr/local/lib/python3.12/dist-packages (from kaggle) (2.32.4)
Requirement already satisfied: setuptools>=21.0.0 in /usr/local/lib/python3.12/dist-packages (from kaggle) (75.2.0)
Requirement already satisfied: six>=1.10 in /usr/local/lib/python3.12/dist-packages (from kaggle) (1.17.0)
Requirement already satisfied: text-unidecode in /usr/local/lib/python3.12/dist-packages (from kaggle) (1.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.12/dist-packages (from kaggle) (4.67.1)
Requirement already satisfied: urllib3>=1.15.1 in /usr/local/lib/python3.12/dist-packages (from kaggle) (2.5.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.12/dist-packages (from kaggle) (0.5.1)
```

```
import os
# If you uploaded kaggle.json to Colab (e.g. via file upload)
os.environ['KAGGLE_CONFIG_DIR'] = "/content"

# or if you place it elsewhere, adjust path accordingly
```

```
!unzip "/content/archive (5).zip" -d /content/data
```

```
Archive:  /content/archive (5).zip
  inflating: /content/data/dataset/covid/01E392EE-69F9-4E33-BFCE-E5C968654078.jpeg
  inflating: /content/data/dataset/covid/1-s2.0-S0140673620303706-fx1_lrg.jpg
  inflating: /content/data/dataset/covid/1-s2.0-S0929664620300449-gr2_lrg-a.jpg
  inflating: /content/data/dataset/covid/1-s2.0-S0929664620300449-gr2_lrg-b.jpg
  inflating: /content/data/dataset/covid/1-s2.0-S0929664620300449-gr2_lrg-c.jpg
  inflating: /content/data/dataset/covid/1-s2.0-S0929664620300449-gr2_lrg-d.jpg
  inflating: /content/data/dataset/covid/1-s2.0-S1684118220300608-main.pdf-001.jpg
  inflating: /content/data/dataset/covid/1-s2.0-S1684118220300608-main.pdf-002.jpg
  inflating: /content/data/dataset/covid/1-s2.0-S1684118220300682-main.pdf-002-a1.png
  inflating: /content/data/dataset/covid/1-s2.0-S1684118220300682-main.pdf-002-a2.png
  inflating: /content/data/dataset/covid/1-s2.0-S1684118220300682-main.pdf-003-b1.png
  inflating: /content/data/dataset/covid/1-s2.0-S1684118220300682-main.pdf-003-b2.png
  inflating: /content/data/dataset/covid/1312A392-67A3-4EBF-9319-810CF6DA5EF6.jpeg
  inflating: /content/data/dataset/covid/1B734A89-A1BF-49A8-A1D3-66FAFA4FAC5D.jpeg
  inflating: /content/data/dataset/covid/23E99E2E-447C-46E5-8EB2-D35D12473C39.png
  inflating: /content/data/dataset/covid/2C26F453-AF3B-4517-BB9E-802CF2179543.jpeg
  inflating: /content/data/dataset/covid/31BA3780-2323-493F-8AED-62081B9C383B.jpeg
  inflating: /content/data/dataset/covid/41591_2020_819_Fig1_HTML.webp-day10.png
  inflating: /content/data/dataset/covid/41591_2020_819_Fig1_HTML.webp-day5.png
  inflating: /content/data/dataset/covid/6CB4EFC6-68FA-4CD5-940C-BEFA8DAFE9A7.jpeg
  inflating: /content/data/dataset/covid/7AF6C1AF-D249-4BD2-8C26-449304105D03.jpeg
  inflating: /content/data/dataset/covid/7C69C012-7479-493F-8722-ABC29C60A2DD.jpeg
  inflating: /content/data/dataset/covid/80446565-E090-4187-A031-9D3CEAA586C8.jpeg
  inflating: /content/data/dataset/covid/85E52EB3-56E9-4D67-82DA-DEA247C82886.jpeg
  inflating: /content/data/dataset/covid/8FDE8DBA-CFBD-4B4C-B1A4-6F36A93B7E87.jpeg
  inflating: /content/data/dataset/covid/93FE0BB1-022D-4F24-9727-987A07975FFB.jpeg
  inflating: /content/data/dataset/covid/9C34AF49-E589-44D5-92D3-168B3B04E4A6.jpeg
  inflating: /content/data/dataset/covid/CD50BA96-6982-4C80-AE7B-5F67ACDBFA56.jpeg
  inflating: /content/data/dataset/covid/E63574A7-4188-4C8D-8D17-9D67A18A1AFA.jpeg
  inflating: /content/data/dataset/covid/F2DE909F-E19C-4900-92F5-8F435B031AC6.jpeg
  inflating: /content/data/dataset/covid/F4341CE7-73C9-45C6-99C8-8567A5484B63.jpeg
  inflating: /content/data/dataset/covid/F63AB6CE-1968-4154-A70F-913AF154F53D.jpeg
  inflating: /content/data/dataset/covid/FE9F9A5D-2830-46F9-851B-1FF4534959BE.jpeg
  inflating: /content/data/dataset/covid/all14238-fig-0001-m-b.jpg
  inflating: /content/data/dataset/covid/all14238-fig-0001-m-c.jpg
  inflating: /content/data/dataset/covid/auntminnie-a-2020_01_28_23_51_6665_2020_01_28_Vietnam_coronavirus.jpeg
  inflating: /content/data/dataset/covid/auntminnie-b-2020_01_28_23_51_6665_2020_01_28_Vietnam_coronavirus.jpeg
  inflating: /content/data/dataset/covid/auntminnie-c-2020_01_28_23_51_6665_2020_01_28_Vietnam_coronavirus.jpeg
  inflating: /content/data/dataset/covid/auntminnie-d-2020_01_28_23_51_6665_2020_01_28_Vietnam_coronavirus.jpeg
  inflating: /content/data/dataset/covid/ciaa199.pdf-001-[illegible]g
  inflating: /content/data/dataset/covid/ciaa199.pdf-001-b.png
  inflating: /content/data/dataset/covid/ciaa199.pdf-001-c.png
  inflating: /content/data/dataset/covid/covid-19-pneumonia-12.jpg
  inflating: /content/data/dataset/covid/covid-19-pneumonia-14-PA.png
  inflating: /content/data/dataset/covid/covid-19-pneumonia-15-PA.jpg
  inflating: /content/data/dataset/covid/covid-19-pneumonia-19.jpg
  inflating: /content/data/dataset/covid/covid-19-pneumonia-2.jpg
  inflating: /content/data/dataset/covid/covid-19-pneumonia-7-PA.jpg
  inflating: /content/data/dataset/covid/gr1_lrg-a.jpg
  inflating: /content/data/dataset/covid/gr1_lrg-b.jpg
  inflating: /content/data/dataset/covid/kjr-21-e24-g001-l-a.jpg
  inflating: /content/data/dataset/covid/kjr-21-e24-g002-l-a.jpg
  inflating: /content/data/dataset/covid/kjr-21-e24-g003-l-a.jpg
  inflating: /content/data/dataset/covid/kjr-21-e25-g001-l-a.jpg
  inflating: /content/data/dataset/covid/lancet-case2a.jpg
  inflating: /content/data/dataset/covid/lancet-case2b.jpg
  inflating: /content/data/dataset/covid/nCoV-radiol.2020200269.fig1-day7.jpeg
```

```
import numpy as np
import matplotlib.pyplot as plt
import os
import cv2 as cv
import random
```

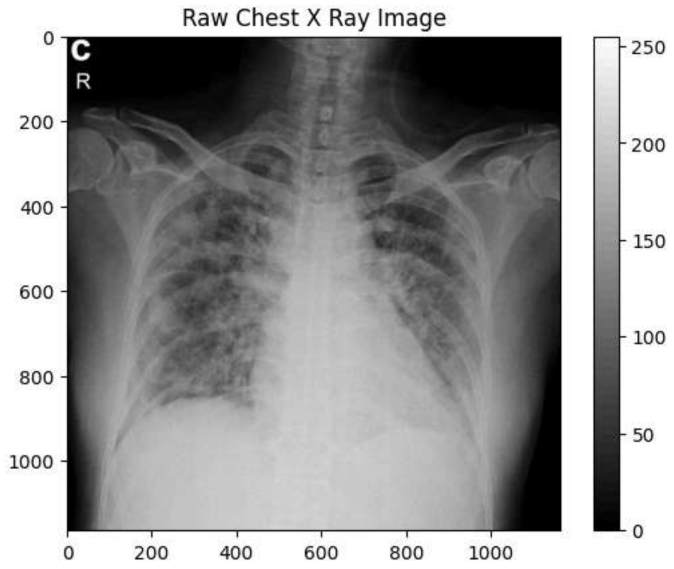
Investigating a single image from the Dataset:

```
# def load_image(path):
#     for img in os.listdir(bacteria_path):
#         print('Image name =',img)
#         image = cv.imread(os.path.join(bacteria_path, img))
#         break

#     return image
```

Investigating single image

```
from keras.preprocessing import image
bacteria_path = '/content/data/dataset/covid/1-s2.0-S0929664620300449-gr2_lrg-c.jpg'

image = cv.imread(bacteria_path)
plt.imshow(image, cmap='gray')
plt.colorbar()
plt.title('Raw Chest X Ray Image')
print(f"The dimensions are {image.shape[0]} pixels height and {image.shape[1]} pixels width")
print(f"The maximum pixel value is {image.max():.4f}")
print(f"The minimum pixel value is {image.min():.4f}")
print(f"The mean value of the pixels is {image.mean():.4f}")
print(f"The standard deviation is {image.std():.4f}")
```

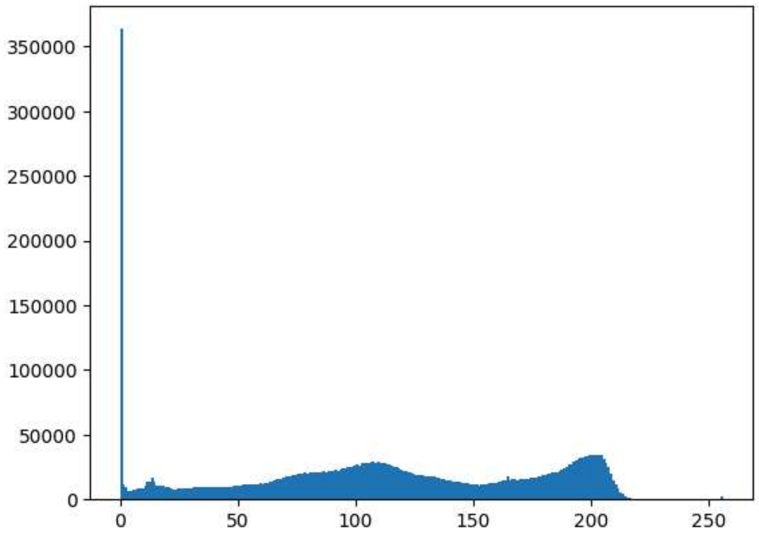
```
The dimensions are 1165 pixels height and 1165 pixels width
The maximum pixel value is 255.0000
The minimum pixel value is 0.0000
The mean value of the pixels is 110.1147
The standard deviation is 64.9676
```



plot histogram

```
plt.hist(image.ravel(),256,[0,256])
plt.show()
```

```
/tmp/ipython-input-3000102515.py:1: MatplotlibDeprecationWarning: Passing the range parameter of hist() positionally is deprecated
  plt.hist(image.ravel(),256,[0,256])
```



Loading images and labels together and resizing images

```
path = '/content/data/dataset'
folders=[]
folders = [f for f in sorted(os.listdir(path))]
print(folders)
```

```
['covid', 'normal']
```

```
labels = folders
print (f'The labels are {labels}')
# setting the size of images that we want
image_size = 256
print(f'All images to be resized into {image_size}*{image_size} pixels')
```

```
The labels are ['covid', 'normal']
All images to be resized into 256*256 pixels
```

```
import cv2 as cv
import os
import numpy as np

def load_train(path, target_size=(128, 128)):
    images = []
    labels = []

    for folder in ['covid', 'normal']:
        folder_path = os.path.join(path, folder)
        for file in os.listdir(folder_path):
            image_path = os.path.join(folder_path, file)
            image = cv.imread(image_path, cv.IMREAD_GRAYSCALE)
            if image is not None:
                # resize to fixed size
                image = cv.resize(image, target_size)
                images.append(image)
                labels.append(0 if folder == 'normal' else 1)
            else:
                print(f"Warning: Could not load {image_path}")

    return np.array(images), np.array(labels)
```

```
path = '/content/data/dataset'
train_images, train_labels = load_train(path, target_size=(128,128))

print(f"Shape of training images = {train_images.shape}")   # (N, 128, 128)
print(f"Shape of training labels = {train_labels.shape}")   # (N,)
```

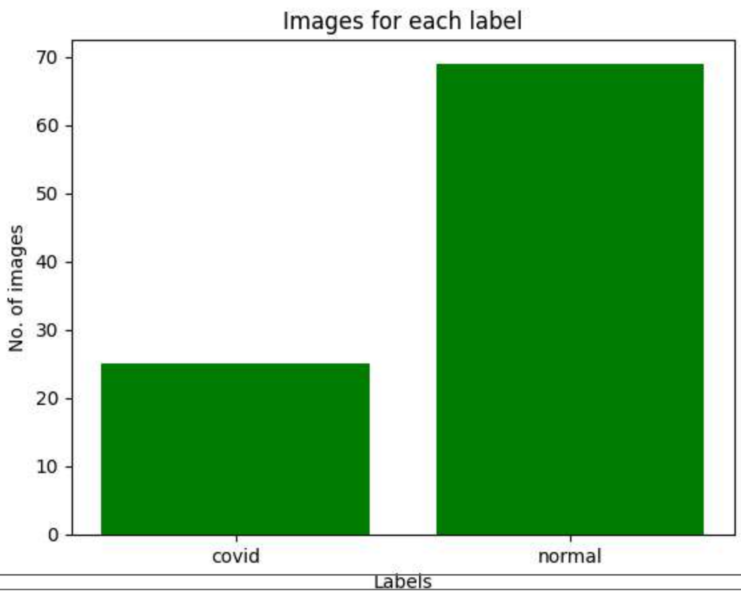
```
Shape of training images = (94, 128, 128)
Shape of training labels = (94,)
```

```
X = train_images
y = train_labels

print(f'Length of X = {len(X)}')
print(f'Length of y = {len(y)}')
```

```
Length of X = 94
Length of y = 94
```

```
# checking the number of images of each class
a = 0
b = 0
for label in y:
    if label == 0:
        a += 1
    if label == 1:
        b += 1
print (f'Number of Normal images = {a}')
print (f'Number of Covid images = {b}')

# plotting the data
x_pos = [i for i, _ in enumerate(labels)]
numbers = [a,b]
plt.bar(x_pos,numbers,color = 'green')
plt.xlabel("Labels")
plt.ylabel("No. of images")
plt.title("Images for each label")
plt.xticks(x_pos, labels)
plt.show()
```

```
Number of Normal images = 25
Number of Covid images = 69
```



```
# Displays images
# Extract 9 random images
print('Display Random Images')
# Adjust the size of your images
plt.figure(figsize=(20,10))
for i in range(9):
    num = random.randint(0,len(X)-1)
    plt.subplot(3, 3, i + 1)
    plt.imshow(X[num],cmap='gray')
    plt.axis('off')
# Adjust subplot parameters to give specified padding
plt.tight_layout()
```

```
Display Random Images
```