**What is Image Classification and Recognition?**

Image Classification and Recognition is a computer vision technique that allows machines to identify and categorize various elements of images and/or videos.  Image recognition models are trained to take an image as input and output one or more labels describing the image.  This core task is a foundational component in solving many computers vision-based machine learning business problems.

**Why is image classification and recognition important?**

Image classification and recognition is one of the most foundational and widely-applicable computer vision task for growth of businesses. From building better content to competing more effectively in industrial market, there are several tips that large and small business can follow to better utilize and benefit from image classification and recognition. This kind of image recognition involving patterns and extraction of useful features has become a building block of many complex computer vision techniques (i.e., object detection, image segmentation, etc.). It also has numerous standalone applications that make it broad and highly-generalizable for artificial intelligence task. The benefits of image classification include but not limited to automated image organization, user-generated content moderation, enhanced visual search, automated photo and video tagging, assistance in security threats, leveraging of hiring processes and identification of variables in images.

**Benefits of Image Classification and Recognition in businesses**

*Incorporation in Automation*

Image processing and machine vision systems play an increasingly important role in quality assurance in an ever-growing number of industries. It is a technology that uses images or image data to identify objects within an image and can recognize, annotate, label, and organize image content. This technology can be used in small and large businesses to recognize images of products, content lists for eCommerce sites, gender classification problem, digital photography, and signs like menu items in restaurants. Small business owners can use image recognition software to automatically populate forms and documents with information about their store, products, and employees without manually typing the data in themselves. This technique is benefited in automated image processing having relative learning curves from data visualization and analysis programs straightforward image editing software’s.

*Assist in Security Threats*

Image recognition software’s are getting better and more useful, and many businesses are taking advantage of it. Of all the AI solutions on the table right now, image recognition software is some of the best in IT industry. Image algorithms using machine learning that helps to understand the cybersecurity threats facing by customers. A simple screenshot can tell a lot about a customer’s situation, and if they need further assistance, image recognition software has made such process even easier. If a business has a new security threat, image recognition allows to run diagnostics without having to compromise system or server.

**Challenges for image classification and recognition**

Different perspectives of one person caught on camera can confuse image classification solutions in capturing digital photography, that can list one person as several individuals. It is important for businesses to provide solutions that consolidate all media into single-person media profiles. Image classification tasks can take hours to process for their multiple categories that need to be referenced into the output solution. Maintaining fast processing speeds that is relevant for time-sensitive investigations as per problems is an important challenge.

**Working of Image Classification and Recognition**

Following example covers a gender-based image classification problem. To construct a model that recognize persons robustly a dataset has been used. The dataset is available at [source.](https://www.kaggle.com/datasets/nipunarora8/age-gender-and-ethnicity-face-data-csv) You can use any image as a test.jpg to see model efficiency.

CELL1:

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import keras

from keras import Sequential

from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import classification\_report

from PIL import Image

import matplotlib.image as mpg

import seaborn as sns

plt.style.use('ggplot')

plt.rcParams['font.family'] = 'sans-serif'

plt.rcParams['font.serif'] = 'Ubuntu'

plt.rcParams['font.monospace'] = 'Ubuntu Mono'

plt.rcParams['font.size'] = 14

plt.rcParams['axes.labelsize'] = 12

plt.rcParams['axes.labelweight'] = 'bold'

plt.rcParams['axes.titlesize'] = 12

plt.rcParams['xtick.labelsize'] = 12

plt.rcParams['ytick.labelsize'] = 12

plt.rcParams['legend.fontsize'] = 12

plt.rcParams['figure.titlesize'] = 12

plt.rcParams['image.cmap'] = 'jet'

plt.rcParams['image.interpolation'] = 'none'

plt.rcParams['figure.figsize'] = (10, 10

                                 )

plt.rcParams['axes.grid']=True

plt.rcParams['lines.linewidth'] = 2

plt.rcParams['lines.markersize'] = 8

colors = ['xkcd:pale orange', 'xkcd:sea blue', 'xkcd:pale red', 'xkcd:sage green', 'xkcd:terra cotta', 'xkcd:dull purple', 'xkcd:teal', 'xkcd: goldenrod', 'xkcd:cadet blue',

'xkcd:scarlet']

bbox\_props = dict(boxstyle="round,pad=0.3", fc=colors[0], alpha=.5)

CELL2:

#upload dataset reference https://www.kaggle.com/datasets/nipunarora8/age-gender-and-ethnicity-face-data-csv

data = pd.read\_csv('age\_gender.csv')

CELL3:

data = data.sample(frac=1).reset\_index().loc[0:1000]

CELL4:

def Convert(string):

    li = list(string.split(" "))

    return li

CELL5:

PIXELS=[]

for i in range(len(data)):

    PIXELS.append(Convert(data.pixels[i]))

NEW\_PIXELS = []

for p in range(len(PIXELS)):

    new\_pixels = []

    for q in range(len(PIXELS[p])):

        new\_pixels.append(int(PIXELS[p][q]))

    NEW\_PIXELS.append(np.array(new\_pixels).reshape((48,48,1)))

CELL6:

images = np.array(NEW\_PIXELS)

data = data.drop(columns=['index','img\_name'])

sns.countplot(data.gender,palette='plasma')

plt.xticks([0,1],['Male','Female'])

plt.grid(True)

CELL7:

data.pixels = NEW\_PIXELS

for i in range(1,5):

    J = np.random.choice(np.arange(0,1000,1))

    plt.subplot(2,2,i)

    plt.title('Label : '+ str(data.gender[J]),fontsize=20)

    plt.imshow(np.array(data.pixels[J]).reshape((48,48)),cmap='gray')

    plt.tight\_layout()

CELL8:

classifier = Sequential()

size = 48

# Step 1 - Convolution

classifier.add(Conv2D(64, (2, 2), input\_shape = (size,size, 1), activation = 'relu'))

classifier.add(MaxPooling2D(pool\_size = (2, 2)))

classifier.add(Conv2D(64, (2, 2), input\_shape = (size,size, 1), activation = 'relu'))

classifier.add(MaxPooling2D(pool\_size = (2, 2)))

classifier.add(Flatten())

#classifier.add(Dense(units = 32, activation = 'relu'))

classifier.add(Dense(units = 1, activation = 'sigmoid'))

# Compiling the CNN

classifier.summary()

CELL9:

labels = np.array(data.gender)

train\_images, test\_images, train\_labels, test\_labels = train\_test\_split(

    images, labels, test\_size=0.15)

classifier.compile(optimizer = 'adam', loss = 'binary\_crossentropy', metrics = ['accuracy'])

history = classifier.fit(train\_images, train\_labels, epochs=10,

                validation\_data=(test\_images, test\_labels),batch\_size=10)

CELL10:

loss = history.history['accuracy']

val\_loss = history.history['val\_accuracy']

plt.plot(np.arange(1,len(loss)+1,1),loss,color='navy', label = 'Accuracy')

plt.plot(np.arange(1,len(loss)+1,1),val\_loss,color='red',label='Validation Accuracy')

plt.legend(fontsize=15)

CELL11:

plt.plot(classifier.predict(test\_images),'.',color='red',label='Predicted Probabilty')

plt.plot(test\_labels,'.',color='navy',label='Actual Labels')

plt.xlabel('Instance Number')

plt.ylabel('Probability')

plt.legend()

CELL12:

predictions = classifier.predict(test\_images)

decision = []

for p in predictions:

    if p>=0.5:

        decision.append(1)

    else:

        decision.append(0)

sns.heatmap(confusion\_matrix(decision,test\_labels),cmap='plasma',annot=True,annot\_kws={"size": 32})

plt.xticks([0.50,1.50],['Male','Female'],fontsize=20)

plt.yticks([0.50,1.50],['Male','Female'],fontsize=20)

CELL13:

clf\_report = classification\_report(decision,test\_labels,

                                   labels=[0,1],

                                   target\_names=['Male','Female'],

                                   output\_dict=True)

plt.subplot(2,1,1)

sns.heatmap(pd.DataFrame(clf\_report).iloc[:-2, :-3].T, annot=True)

plt.subplot(2,1,2)

sns.heatmap(pd.DataFrame(clf\_report).iloc[:-2, 3:].T, annot=True)

plt.tight\_layout()

CELL14:

def rgb2gray(rgb):

    return np.dot(rgb[...,:3], [0.2989, 0.5870, 0.1140])

def ImagePreProcessing(image\_path):

    img = Image.open(image\_path)

    img = img.resize((48,48))

    img\_array = np.asarray(img)

    baw\_img = rgb2gray(img\_array).astype(int)

    final\_img = baw\_img.reshape((48,48))

    return final\_img

plt.imshow(ImagePreProcessing('test.jpg'),cmap='gray')

CELL15:

image\_test = ImagePreProcessing('test.jpg')

prob = 1-classifier.predict(np.array([image\_test]))[0][0]

print("I'm a man, and the classifier says that I'm a man with probability %.2f" %(prob\*100) + '%')