**CODING:**

**#IMPORTING REQUIRED LIBRARIES**

import matplotlib.pyplot as plt import seaborn as sns

import os import gc

from sklearn.model\_selection import train\_test\_split import tensorflow as tf

from tqdm.autonotebook import tqdm import numpy as np

import pandas as pd

from keras import Sequential

from keras.callbacks import EarlyStopping from keras.optimizers import Adam, SGD

from keras.callbacks import ReduceLROnPlateau

keras.layersimport Flatten,Dense,BatchNormalization,Activation,Dropout

fromkeras.layersimportLambda,Input, GlobalAveragePooling2D,BatchNormalization from keras.utils import to\_categorical

# from keras import regularizers

from tensorflow.keras.models import Model from keras.preprocessing.image import load\_img

# from keras.preprocessing.image import img\_to\_array

# from keras.applications.imagenet\_utils import decode\_predictions

**#READING LABELS CSV FILE**

labels = pd.read\_csv('/kaggle/input/dog-breed-identification/labels.csv')

labels.head() plt.show()

**#FUNCTION TO SHOW BAR LENGTH**

def barw(ax):

for p in ax.patches:

val = p.get\_width() #height of the bar

x = p.get\_x()+ p.get\_width() # x- position

y = p.get\_y() + p.get\_height()/2 #y-position ax.annotate(round(val,2),(x,y))

**#FINDING TOP DOG BRANDS**

plt.figure(figsize = (15,30)) ax

=sns.countplot(y=labels['breed'],order=labels['breed'].value\_counts().ind ex)

barw(ax0) plt.show()

**#CHECKING ONE IMAGE**

from IPython.display import display, Image

Image("../input/dog-breed identification/test/002c2a3117c2193b4d26400ce431eebd.jpg")

**#CREATE LIST OF ALPHABETICALLY SORTED LABELS**.

classes = sorted(list(set(labels['breed']))) n\_classes = len(classes)

print('Total unique breed {}'.format(n\_classes))

**#MAP EACH LABEL STRING TO AN INTEGER LABEL**.

class\_to\_num = dict(zip(classes, range(n\_classes))) class\_to\_num

input\_shape = (331,331,3)

def images\_to\_array(directory, label\_dataframe, target\_size = input\_shape):

image\_labels = label\_dataframe['breed']

images = np.zeros([len(label\_dataframe), target\_size[0], target\_size[1], target\_size[2]],dtype=np.uint8) #as we have huge data and limited ram memory. uint8 takes less memory

y = np.zeros([len(label\_dataframe),1],dtype = np.uint8)

for ix, image\_name in enumerate(tqdm(label\_dataframe['id'].values)): img\_dir = os.path.join(directory, image\_name + '.jpg')

img = load\_img(img\_dir, target\_size = target\_size) #img = np.expand\_dims(img, axis=0)

#img = processed\_image\_resnet(img) #img = img/255

images[ix]=img

images[ix] = img\_to\_array(img) del img

dog\_breed = image\_labels[ix] y[ix] = class\_to\_num[dog\_breed] y = to\_categorical(y)

return images,y

import os if

len(os.listdir('/kaggle/input/dog-breed-identification/train/'))== len(labels['id']):

print('Number of file matches number of actual images!') else:

print('Number of file doesnot matches number of actual images!!')

**#PROGRAM RUNNING TIME**

import time

t = time.time()

X,y=images\_to\_array('/kaggle/input/dog-breed-identification/train', labels[:])

print('runtime in seconds: {}'.format(time.time() - t)) # np.where(y[5]==1)[0][0]

**# LETS CHECK SOME DOGS AND THEIR BREEDS**

n=25

**# SETUP THE FIGURE**

plt.figure(figsize=(30,30))

for i in range(n):

# print(i)

ax = plt.subplot(5, 5, i+1) plt.title(classes[np.where(y[i] ==1)[0][0]])

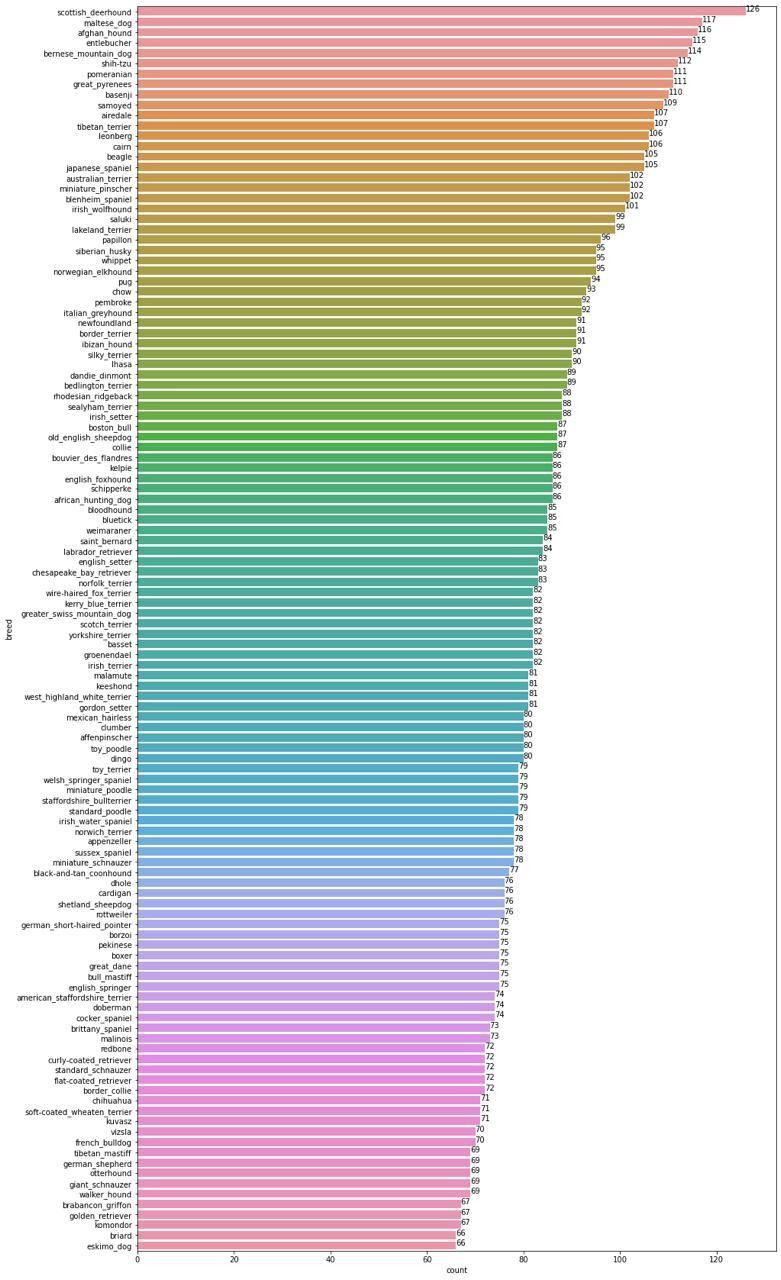
plt.imshow(X[i].astype('int32')) # .astype('int32') ---> as imshow() needs integer data to read the image

# RESULT:

LABELS:

|  |  |  |
| --- | --- | --- |
| **id** | **breed** |  |
| **0** | 000bec180eb18c7604dcecc8fe0dba07 | boston\_bull |
| **1** | 001513dfcb2ffafc82cccf4d8bbaba97 | dingo |
| **2** | 001cdf01b096e06d78e9e5112d419397 | pekinese |
| **3** | 00214f311d5d2247d5dfe4fe24b2303d | bluetick |
| **4** | 0021f9ceb3235effd7fcde7f7538ed62 | golden\_retriever |
| **id** | **breed** |  |
| **count** | 10222 | 10222 |
| **unique** | 10222 | 120 |
| **top** | f589e77c9b0facc8ece5f07e3c04fc46 | scottish\_deerhound |
| **freq** | 1 | 126 |

BARGRAPH:



CHECK ONE IMAGE FROM DATASET:



TOTAL NO OF BREEDS:

Total unique breed 120

{'affenpinscher': 0,

'afghan\_hound': 1,

'african\_hunting\_dog': 2,

'airedale': 3,

'american\_staffordshire\_terrier': 4,

'appenzeller': 5,

'australian\_terrier': 6,

'basenji': 7,

'basset': 8,

'beagle': 9,

'bedlington\_terrier': 10,

'bernese\_mountain\_dog': 11,

'black-and-tan\_coonhound': 12,

'blenheim\_spaniel': 13,

'bloodhound': 14,

'bluetick': 15,

'border\_collie': 16,

'border\_terrier': 17,

'borzoi': 18,

'boston\_bull': 19,

'bouvier\_des\_flandres': 20,

'boxer': 21,

'brabancon\_griffon': 22,

'briard': 23,

'brittany\_spaniel': 24,

'bull\_mastiff': 25,

'cairn': 26,

'cardigan': 27,

'chesapeake\_bay\_retriever': 28,

'chihuahua': 29,

'chow': 30,

'clumber': 31,

'cocker\_spaniel': 32,

'collie': 33,

'curly-coated\_retriever': 34,

'dandie\_dinmont': 35,

'dhole': 36,

'dingo': 37,

'doberman': 38,

'english\_foxhound': 39,

'english\_setter': 40,

'english\_springer': 41,

'entlebucher': 42,

'eskimo\_dog': 43,

'flat-coated\_retriever': 44,

'french\_bulldog': 45,

'german\_shepherd': 46,

'german\_short-haired\_pointer': 47,

'giant\_schnauzer': 48,

'golden\_retriever': 49,

'gordon\_setter': 50,

'great\_dane': 51,

'great\_pyrenees': 52,

'greater\_swiss\_mountain\_dog': 53,

'groenendael': 54,

'ibizan\_hound': 55,

'irish\_setter': 56,

'irish\_terrier': 57,

'irish\_water\_spaniel': 58,

'irish\_wolfhound': 59,

'italian\_greyhound': 60,

'japanese\_spaniel': 61,

'keeshond': 62,

'kelpie': 63,

'kerry\_blue\_terrier': 64,

'komondor': 65,

'kuvasz': 66,

'labrador\_retriever': 67,

'lakeland\_terrier': 68,

'leonberg': 69,

'lhasa': 70,

'malamute': 71,

'malinois': 72,

'maltese\_dog': 73,

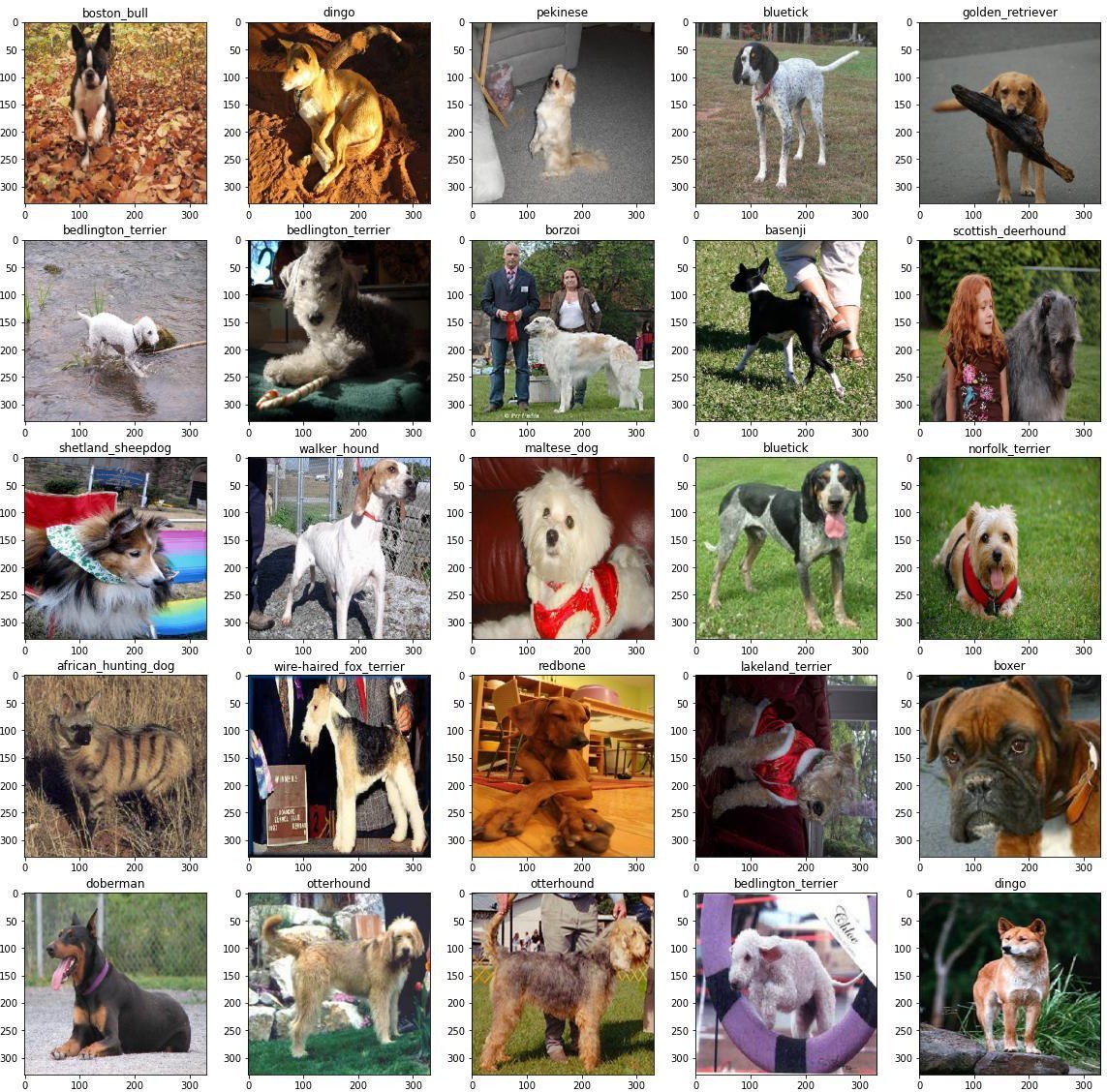
'mexican\_hairless': 74,

'miniature\_pinscher': 75,

'miniature\_poodle': 76,

'miniature\_schnauzer': 77,

**DOG BREED PREDICTION**:



# CONCLUSION :

Dog breed prediction using deep learning is developed using convolutional neural network to predict the breed of dogs by taking their images as input. This system is easy and convenient for users to predict the dog's breed. The system would predict the dog breeds at a much higher accuracy . AWe have demonstrated modified approach of the state of art network.

**FUTURE WORK:**

Future work should further explore the potential of convolutional neural networks in dog breed prediction. Given the success of our keypoint detection network, this is a promising technique for future projects. That said, neural networks take an enormous time to train and we were unable to perform many iterations on our technique due to time constraints. We recommend further exploration into neural networks for keypoint detection, specifically by training networks with a different architecture and batch iterator to see what approaches might have greater success. Also, given our success with neural networks and keypoint detection, we recommend implementing a neural network for breed classification as well since this has not been performed in the literature. We were unable to experiment with this approach due to the time constraints of neural networks but believe that they would match if not improve upon our classification results. Ultimately, neural networks are time consuming to train and iterate upon, which should be kept in consideration for future efforts; still, neural networks are formidable classifiers that will increase pred