Assignment3NewDir

December 5, 2022

[269]:
```
import seaborn as sb
import matplotlib.pyplot as plt
import warnings
import pandas as pd
import numpy as np
import requests
import io

import torch
import torch.nn as nn
import torch.nn.functional as F
from torch.utils.data import DataLoader
from torchvision import datasets, transforms, models
from torchvision.utils import make_grid

from PIL import Image
from IPython.display import display
import cv2
from PIL import ImageFile
import torchvision.transforms as transforms
ImageFile.LOAD_TRUNCATED_IMAGES = True

import glob
import os
import random
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')

if(True):
  #link to google drive
  from google.colab import drive
  drive.mount('/content/drive')

if(True):
  #unzip data
  import zipfile
```

```
path_to_zip_file = r'/content/drive/MyDrive/College Junior/CS 4375/
↪dog-breed-identification.zip'
directory_to_extract_to = "dog-breed-identification"
!mkdir "dog-breed-identification"
with zipfile.ZipFile(path_to_zip_file, 'r') as zip_ref:
    zip_ref.extractall(directory_to_extract_to)
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
mkdir: cannot create directory ‘dog-breed-identification’: File exists
```

This dataset is from a Kaggle machine learning competition several years ago and includes hundreds of dog pictures to identify. There are 120 possible dog breeds in the target class and the images have three color channels and inconsistent resolutions, so they must be reshaped and flattened before training.

[270]:
```
%cd "/content/dog-breed-identification"

labels = pd.read_csv('labels.csv')
labelnames = pd.read_csv('sample_submission.csv')

print(f"There are {len(labelnames)} label names")
print(labelnames.head(5).to_string())
print()
print(f"There are {len(labels)} labels")
print(labels.head(5).to_string())
# from the looks of the csv file;, it is
# given a submission [identified by id], gives probabilities [in the columns]␣
↪of being [column title]
```

```
/content/dog-breed-identification
There are 10357 label names
                                 id  affenpinscher  afghan_hound
african_hunting_dog  airedale  american_staffordshire_terrier  appenzeller
australian_terrier   basenji    basset    beagle  bedlington_terrier
bernese_mountain_dog  black-and-tan_coonhound  blenheim_spaniel  bloodhound
bluetick  border_collie  border_terrier    borzoi  boston_bull
bouvier_des_flandres     boxer  brabancon_griffon    briard  brittany_spaniel
bull_mastiff     cairn  cardigan  chesapeake_bay_retriever  chihuahua      chow
clumber  cocker_spaniel    collie  curly-coated_retriever  dandie_dinmont
dhole     dingo  doberman  english_foxhound  english_setter  english_springer
entlebucher  eskimo_dog  flat-coated_retriever  french_bulldog  german_shepherd
german_short-haired_pointer  giant_schnauzer  golden_retriever  gordon_setter
great_dane  great_pyrenees  greater_swiss_mountain_dog  groenendael
ibizan_hound  irish_setter  irish_terrier  irish_water_spaniel  irish_wolfhound
italian_greyhound  japanese_spaniel  keeshond    kelpie  kerry_blue_terrier
komondor    kuvasz  labrador_retriever  lakeland_terrier  leonberg     lhasa
malamute  malinois  maltese_dog  mexican_hairless  miniature_pinscher
```

```
miniature_poodle  miniature_schnauzer  newfoundland  norfolk_terrier
norwegian_elkhound  norwich_terrier  old_english_sheepdog  otterhound  papillon
pekinese  pembroke  pomeranian       pug   redbone  rhodesian_ridgeback
rottweiler  saint_bernard    saluki   samoyed  schipperke  scotch_terrier
scottish_deerhound  sealyham_terrier  shetland_sheepdog  shih-tzu
siberian_husky  silky_terrier  soft-coated_wheaten_terrier
staffordshire_bullterrier  standard_poodle  standard_schnauzer  sussex_spaniel
tibetan_mastiff  tibetan_terrier  toy_poodle  toy_terrier    vizsla
walker_hound  weimaraner  welsh_springer_spaniel  west_highland_white_terrier
whippet  wire-haired_fox_terrier  yorkshire_terrier
0  000621fb3cbb32d8935728e48679680e       0.008333      0.008333
0.008333  0.008333                          0.008333     0.008333
0.008333  0.008333  0.008333  0.008333           0.008333              0.008333
0.008333          0.008333    0.008333  0.008333       0.008333        0.008333
0.008333     0.008333              0.008333  0.008333           0.008333
0.008333          0.008333      0.008333  0.008333  0.008333
0.008333   0.008333  0.008333  0.008333        0.008333  0.008333
0.008333        0.008333  0.008333  0.008333  0.008333          0.008333
0.008333          0.008333     0.008333    0.008333               0.008333
0.008333         0.008333                     0.008333         0.008333
0.008333       0.008333    0.008333        0.008333                    0.008333
0.008333      0.008333      0.008333       0.008333             0.008333
0.008333           0.008333          0.008333  0.008333  0.008333
0.008333  0.008333  0.008333           0.008333          0.008333  0.008333
0.008333  0.008333  0.008333     0.008333          0.008333             0.008333
0.008333             0.008333      0.008333         0.008333             0.008333
0.008333              0.008333    0.008333  0.008333  0.008333  0.008333
0.008333  0.008333  0.008333             0.008333    0.008333       0.008333
0.008333  0.008333    0.008333        0.008333             0.008333
0.008333           0.008333  0.008333        0.008333       0.008333
0.008333                    0.008333         0.008333            0.008333
0.008333         0.008333         0.008333    0.008333     0.008333  0.008333
0.008333    0.008333                 0.008333                     0.008333
0.008333                  0.008333           0.008333
1  00102ee9d8eb90812350685311fe5890       0.008333      0.008333
0.008333  0.008333                          0.008333     0.008333
0.008333  0.008333  0.008333  0.008333           0.008333              0.008333
0.008333          0.008333    0.008333  0.008333       0.008333        0.008333
0.008333     0.008333              0.008333  0.008333           0.008333
0.008333          0.008333      0.008333  0.008333  0.008333
0.008333   0.008333  0.008333  0.008333        0.008333  0.008333
0.008333        0.008333  0.008333  0.008333  0.008333          0.008333
0.008333          0.008333     0.008333    0.008333               0.008333
0.008333         0.008333                     0.008333         0.008333
0.008333       0.008333    0.008333        0.008333                    0.008333
0.008333      0.008333      0.008333       0.008333             0.008333
0.008333           0.008333          0.008333  0.008333  0.008333
0.008333  0.008333  0.008333           0.008333          0.008333  0.008333
```

0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333
2 0012a730dfa437f5f3613fb75efcd4ce 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333
3 001510bc8570bbeee98c8d80c8a95ec1 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333
0.008333 0.008333 0.008333 0.008333 0.008333 0.008333

```
0.008333  0.008333  0.008333     0.008333          0.008333            0.008333
0.008333             0.008333      0.008333         0.008333             0.008333
0.008333               0.008333    0.008333  0.008333  0.008333  0.008333
0.008333  0.008333  0.008333             0.008333    0.008333       0.008333
0.008333  0.008333    0.008333        0.008333            0.008333
0.008333           0.008333  0.008333        0.008333       0.008333
0.008333                    0.008333         0.008333            0.008333
0.008333         0.008333         0.008333    0.008333     0.008333  0.008333
0.008333    0.008333                 0.008333                      0.008333
0.008333                  0.008333           0.008333
4  001a5f3114548acdefa3d4da05474c2e       0.008333      0.008333
0.008333  0.008333                          0.008333     0.008333
0.008333  0.008333  0.008333  0.008333            0.008333              0.008333
0.008333          0.008333    0.008333  0.008333       0.008333        0.008333
0.008333     0.008333              0.008333  0.008333           0.008333
0.008333          0.008333      0.008333  0.008333  0.008333
0.008333   0.008333  0.008333  0.008333        0.008333  0.008333
0.008333        0.008333  0.008333  0.008333  0.008333          0.008333
0.008333          0.008333     0.008333    0.008333               0.008333
0.008333         0.008333                      0.008333         0.008333
0.008333       0.008333    0.008333        0.008333                    0.008333
0.008333      0.008333      0.008333       0.008333             0.008333
0.008333           0.008333          0.008333  0.008333  0.008333
0.008333  0.008333  0.008333           0.008333          0.008333  0.008333
0.008333  0.008333  0.008333     0.008333          0.008333            0.008333
0.008333             0.008333      0.008333         0.008333             0.008333
0.008333               0.008333    0.008333  0.008333  0.008333  0.008333
0.008333  0.008333  0.008333             0.008333    0.008333       0.008333
0.008333  0.008333    0.008333        0.008333            0.008333
0.008333           0.008333  0.008333        0.008333       0.008333
0.008333                    0.008333         0.008333            0.008333
0.008333         0.008333         0.008333    0.008333     0.008333  0.008333
0.008333    0.008333                 0.008333                      0.008333
0.008333                  0.008333           0.008333

There are 10222 labels
                                 id              breed
0  000bec180eb18c7604dcecc8fe0dba07       boston_bull
1  001513dfcb2ffafc82cccf4d8bbaba97             dingo
2  001cdf01b096e06d78e9e5112d419397          pekinese
3  00214f311d5d2247d5dfe4fe24b2303d          bluetick
4  0021f9ceb3235effd7fcde7f7538ed62  golden_retriever
```

[271]:
```
#print test images
#this cell is a testing cell
if(False):
  %cd "train"
```

```
  fileDir = "." #'dog-breed-identification/train'
  listOfCuteDogImages = [os.path.join(fileDir, _) for _ in os.listdir(fileDir)]
  #print(listOfCuteDogImages[0:3]) #print first 3 image filename

  for imageName in listOfCuteDogImages[7:11]: #plot images of 7th to 11th dog
    print (os.path.join(imageName))
    img = cv2.imread(os.path.join(imageName))
    #print(img) #this shows numbers..
    img_cvt=cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    plt.imshow(img_cvt)
    plt.show()
  %cd ..
```

[272]:
```
codes = range(len(labelnames))
breed_to_code = dict(zip(labelnames, codes))
code_to_breed = dict(zip(codes, labelnames))
labels['target'] =  [breed_to_code[x] for x in labels.breed]
labels['rank'] = labels.groupby('breed').rank()['id']
#labels = labels[0:1000] #testing
labels_pivot = labels.pivot('id', 'breed', 'target').reset_index().fillna(0)

#split into training samples and testing samples (valid)
train = labels_pivot.sample(frac=0.85)
valid = labels_pivot[~labels_pivot['id'].isin(train['id'])]
print(train.shape, valid.shape)

n_epochs = 8
```

```
(8689, 121) (1533, 121)
```

[273]:
```
import pickle
from os.path import exists
pickleFilname = f'/content/drive/MyDrive/College Junior/CS 4375/
 ↪model_transfer_{train.shape[0]}images{n_epochs}epochs.pkl'

if( exists(pickleFilname ) ):
    with open(pickleFilname , 'rb') as f:
        model_transfer= pickle.load(f)
```

0.0.1 Data

[274]:
```
print(train.head(5).to_string())
listOfTrain = list(train['id']) #list training data, without ".jpg"
```

```
breed                                id  affenpinscher  afghan_hound
african_hunting_dog  airedale  american_staffordshire_terrier  appenzeller
```

```
australian_terrier  basenji  basset  beagle  bedlington_terrier
bernese_mountain_dog  black-and-tan_coonhound  blenheim_spaniel  bloodhound
bluetick  border_collie  border_terrier  borzoi  boston_bull
bouvier_des_flandres  boxer  brabancon_griffon  briard  brittany_spaniel
bull_mastiff  cairn  cardigan  chesapeake_bay_retriever  chihuahua  chow
clumber  cocker_spaniel  collie  curly-coated_retriever  dandie_dinmont  dhole
dingo  doberman  english_foxhound  english_setter  english_springer  entlebucher
eskimo_dog  flat-coated_retriever  french_bulldog  german_shepherd
german_short-haired_pointer  giant_schnauzer  golden_retriever  gordon_setter
great_dane  great_pyrenees  greater_swiss_mountain_dog  groenendael
ibizan_hound  irish_setter  irish_terrier  irish_water_spaniel  irish_wolfhound
italian_greyhound  japanese_spaniel  keeshond  kelpie  kerry_blue_terrier
komondor  kuvasz  labrador_retriever  lakeland_terrier  leonberg  lhasa
malamute  malinois  maltese_dog  mexican_hairless  miniature_pinscher
miniature_poodle  miniature_schnauzer  newfoundland  norfolk_terrier
norwegian_elkhound  norwich_terrier  old_english_sheepdog  otterhound  papillon
pekinese  pembroke  pomeranian  pug  redbone  rhodesian_ridgeback  rottweiler
saint_bernard  saluki  samoyed  schipperke  scotch_terrier  scottish_deerhound
sealyham_terrier  shetland_sheepdog  shih-tzu  siberian_husky  silky_terrier
soft-coated_wheaten_terrier  staffordshire_bullterrier  standard_poodle
standard_schnauzer  sussex_spaniel  tibetan_mastiff  tibetan_terrier  toy_poodle
toy_terrier  vizsla  walker_hound  weimaraner  welsh_springer_spaniel
west_highland_white_terrier  whippet  wire-haired_fox_terrier  yorkshire_terrier
5657   8e802d5459b5e857e832c721d6cd3a35            0.0           0.0
0.0       0.0                                0.0          0.0                 0.0
0.0     0.0     0.0                  0.0                    0.0
0.0              14.0         0.0       0.0           0.0             0.0
0.0          0.0                    0.0    0.0                0.0     0.0
0.0           0.0    0.0       0.0                         0.0        0.0   0.0
0.0             0.0     0.0                      0.0             0.0    0.0
0.0       0.0               0.0             0.0               0.0          0.0
0.0                      0.0             0.0              0.0
0.0              0.0               0.0            0.0         0.0
0.0                          0.0          0.0           0.0           0.0
0.0                   0.0              0.0                0.0               0.0
0.0     0.0                  0.0       0.0     0.0                 0.0
0.0       0.0    0.0       0.0       0.0          0.0               0.0
0.0               0.0                  0.0           0.0             0.0
0.0              0.0                   0.0         0.0       0.0       0.0
0.0         0.0  0.0      0.0                   0.0         0.0            0.0
0.0      0.0         0.0             0.0                0.0              0.0
0.0       0.0             0.0            0.0                            0.0
0.0              0.0                 0.0             0.0              0.0
0.0         0.0          0.0     0.0           0.0         0.0
0.0                          0.0      0.0                       0.0
0.0
6577   a57247bc5d572abd95f1aea215b7de77            0.0           0.0
0.0       0.0                                0.0          0.0                 0.0
```

```
0.0     0.0     0.0                 0.0                   0.0
0.0               0.0         0.0       0.0            0.0             0.0
0.0          0.0                    0.0    0.0               0.0     0.0
0.0           0.0    0.0       0.0                        0.0        0.0   0.0
0.0             0.0     0.0                      0.0             0.0    0.0
0.0       0.0               0.0             0.0               0.0          0.0
0.0                     0.0             0.0              0.0
0.0              0.0               0.0            0.0         0.0
0.0                         0.0          0.0           0.0           0.0
0.0                   0.0              0.0               0.0               0.0
0.0     0.0                 0.0       0.0     0.0                 0.0
0.0       0.0    0.0       0.0       0.0          0.0               0.0
0.0              0.0                  0.0          0.0             0.0
0.0              0.0                  0.0         0.0       0.0       0.0
0.0         0.0  0.0      0.0                  0.0         0.0            0.0
0.0      0.0          0.0             0.0                 0.0               0.0
0.0     101.0             0.0            0.0                           0.0
0.0              0.0                 0.0             0.0             0.0
0.0         0.0          0.0     0.0           0.0         0.0
0.0                          0.0      0.0                     0.0
0.0
7455   bb4aa91f61fa6e06dbc1a3869738b889            0.0           0.0
0.0       0.0                              0.0          0.0                 0.0
0.0     0.0     0.0                 0.0                   0.0
0.0               0.0         0.0       0.0            0.0             0.0
0.0          0.0                    0.0    0.0               0.0     0.0
0.0           0.0    0.0       0.0                        0.0        0.0   0.0
0.0             0.0     0.0                      0.0             0.0    0.0
0.0       0.0               0.0             0.0               0.0          0.0
0.0                     0.0             0.0              0.0
0.0              0.0               0.0            0.0         0.0
0.0                         0.0          0.0           0.0           0.0
0.0                   0.0              0.0               0.0               0.0
0.0     0.0                65.0       0.0     0.0                 0.0
0.0       0.0    0.0       0.0       0.0          0.0               0.0
0.0              0.0                  0.0          0.0             0.0
0.0              0.0                  0.0         0.0       0.0       0.0
0.0         0.0  0.0      0.0                  0.0         0.0            0.0
0.0      0.0          0.0             0.0                 0.0               0.0
0.0       0.0             0.0            0.0                           0.0
0.0              0.0                 0.0             0.0             0.0
0.0         0.0          0.0     0.0           0.0         0.0
0.0                          0.0      0.0                     0.0
0.0
4552   71bd232f6f89bf4ea89af19e63a75378            0.0           0.0
0.0       0.0                              0.0          0.0                 0.0
0.0     0.0     0.0                 0.0                   0.0
0.0               0.0         0.0       0.0            0.0             0.0
```

```
0.0          0.0                   0.0    0.0                0.0     0.0
0.0           0.0    0.0       0.0                        0.0        0.0   0.0
0.0             0.0     0.0                     0.0            0.0    0.0
0.0       0.0               0.0             0.0               0.0          0.0
0.0                    0.0             0.0              0.0
0.0             0.0               0.0            0.0         0.0
0.0                        0.0          0.0           0.0           0.0
0.0                   0.0              0.0                0.0               0.0
0.0     0.0                  0.0       0.0     0.0                 0.0
0.0       0.0    0.0       0.0       0.0          0.0               0.0
0.0              0.0                  0.0          79.0             0.0
0.0             0.0                   0.0         0.0       0.0       0.0
0.0         0.0  0.0      0.0                  0.0         0.0           0.0
0.0      0.0          0.0             0.0                 0.0               0.0
0.0       0.0             0.0            0.0                           0.0
0.0             0.0                 0.0             0.0              0.0
0.0         0.0          0.0     0.0           0.0         0.0
0.0                          0.0      0.0                      0.0
0.0
9234   e7fda54b5e96c6048dad68414568f142            0.0           0.0
0.0       0.0                                0.0          0.0                 0.0
0.0     0.0     0.0                  0.0                   0.0
0.0               0.0         0.0       0.0            0.0             0.0
0.0          0.0                     0.0    0.0               0.0     0.0
0.0           0.0    0.0       0.0                        0.0        0.0   0.0
0.0             0.0     0.0                     0.0            0.0    0.0
0.0       0.0               0.0             0.0               0.0          0.0
0.0                    0.0             0.0              0.0
0.0             0.0               0.0            0.0         0.0
0.0                        0.0          0.0           0.0           0.0
0.0                   0.0              0.0                0.0               0.0
0.0     0.0                  0.0       0.0     0.0                 0.0
0.0       0.0    0.0       0.0       0.0          0.0               0.0
0.0              0.0                  0.0          0.0              0.0
0.0             0.0                   0.0         0.0       0.0       0.0
0.0         0.0  0.0      0.0                  0.0         0.0           0.0
0.0      0.0          0.0             0.0                 0.0               0.0
0.0       0.0             0.0            0.0                           0.0
0.0             0.0                 0.0             0.0              0.0
0.0         0.0          0.0     0.0           0.0         0.0
0.0                        117.0      0.0                      0.0
0.0
```

[275]:
```
print(valid.head(5).to_string())
listOfTest = list(valid['id']) #list of testing data, without ".jpg"
```

```
breed                              id  affenpinscher  afghan_hound
african_hunting_dog  airedale  american_staffordshire_terrier  appenzeller
```

```
australian_terrier  basenji  basset  beagle  bedlington_terrier
bernese_mountain_dog  black-and-tan_coonhound  blenheim_spaniel  bloodhound
bluetick  border_collie  border_terrier  borzoi  boston_bull
bouvier_des_flandres  boxer  brabancon_griffon  briard  brittany_spaniel
bull_mastiff  cairn  cardigan  chesapeake_bay_retriever  chihuahua  chow
clumber  cocker_spaniel  collie  curly-coated_retriever  dandie_dinmont  dhole
dingo  doberman  english_foxhound  english_setter  english_springer  entlebucher
eskimo_dog  flat-coated_retriever  french_bulldog  german_shepherd
german_short-haired_pointer  giant_schnauzer  golden_retriever  gordon_setter
great_dane  great_pyrenees  greater_swiss_mountain_dog  groenendael
ibizan_hound  irish_setter  irish_terrier  irish_water_spaniel  irish_wolfhound
italian_greyhound  japanese_spaniel  keeshond  kelpie  kerry_blue_terrier
komondor  kuvasz  labrador_retriever  lakeland_terrier  leonberg  lhasa
malamute  malinois  maltese_dog  mexican_hairless  miniature_pinscher
miniature_poodle  miniature_schnauzer  newfoundland  norfolk_terrier
norwegian_elkhound  norwich_terrier  old_english_sheepdog  otterhound  papillon
pekinese  pembroke  pomeranian  pug  redbone  rhodesian_ridgeback  rottweiler
saint_bernard  saluki  samoyed  schipperke  scotch_terrier  scottish_deerhound
sealyham_terrier  shetland_sheepdog  shih-tzu  siberian_husky  silky_terrier
soft-coated_wheaten_terrier  staffordshire_bullterrier  standard_poodle
standard_schnauzer  sussex_spaniel  tibetan_mastiff  tibetan_terrier  toy_poodle
toy_terrier  vizsla  walker_hound  weimaraner  welsh_springer_spaniel
west_highland_white_terrier  whippet  wire-haired_fox_terrier  yorkshire_terrier
5      002211c81b498ef88e1b40b9abf84e1d            0.0           0.0
0.0       0.0                                0.0          0.0                 0.0
0.0     0.0     0.0                11.0                    0.0
0.0               0.0         0.0       0.0           0.0             0.0
0.0          0.0                    0.0    0.0                0.0     0.0
0.0           0.0    0.0       0.0                         0.0        0.0   0.0
0.0             0.0     0.0                      0.0             0.0    0.0
0.0       0.0               0.0             0.0               0.0          0.0
0.0                      0.0             0.0              0.0
0.0              0.0               0.0            0.0         0.0
0.0                          0.0          0.0           0.0           0.0
0.0                   0.0              0.0                0.0               0.0
0.0     0.0                  0.0       0.0     0.0                 0.0
0.0       0.0    0.0       0.0       0.0          0.0               0.0
0.0               0.0                  0.0           0.0             0.0
0.0              0.0                   0.0         0.0       0.0       0.0
0.0         0.0  0.0      0.0                   0.0         0.0            0.0
0.0      0.0         0.0             0.0                0.0              0.0
0.0       0.0             0.0            0.0                            0.0
0.0              0.0                 0.0             0.0              0.0
0.0         0.0          0.0     0.0           0.0         0.0
0.0                          0.0      0.0                       0.0
0.0
14     0075dc49dab4024d12fafe67074d8a81            0.0           0.0
0.0       0.0                                0.0          0.0                 0.0
```

```
0.0     0.0     0.0                 0.0                   0.0
0.0               0.0         0.0       0.0            0.0             0.0
0.0          0.0                    0.0    0.0               0.0     0.0
0.0           0.0    0.0       0.0                         0.0        0.0   0.0
0.0             0.0     0.0                       0.0             0.0    0.0
0.0       0.0               0.0             0.0               0.0          0.0
0.0                     0.0             0.0              0.0
0.0              0.0               0.0            0.0         0.0
0.0                         0.0          0.0           0.0           0.0
0.0                   0.0             0.0               0.0               0.0
0.0     0.0                 0.0       0.0     0.0                 0.0
0.0       0.0    0.0       0.0       0.0          0.0               0.0
0.0              0.0                   0.0          0.0             80.0
0.0              0.0                   0.0         0.0       0.0       0.0
0.0         0.0  0.0      0.0                   0.0         0.0            0.0
0.0      0.0          0.0             0.0                 0.0               0.0
0.0       0.0             0.0            0.0                           0.0
0.0              0.0                 0.0             0.0              0.0
0.0         0.0          0.0     0.0           0.0         0.0
0.0                         0.0      0.0                     0.0
0.0
26      00a862390341c5be090dd72bd2bc19ef            0.0           0.0
0.0       0.0                               0.0          0.0                 0.0
0.0     0.0     0.0                 0.0                   0.0
0.0               0.0         0.0       0.0            0.0             0.0
0.0          0.0                    0.0    0.0               0.0     0.0
0.0           0.0    0.0       0.0                         0.0        0.0   0.0
0.0             0.0     0.0                       0.0             0.0    0.0
0.0       0.0               0.0             0.0               0.0          0.0
0.0                     0.0             0.0              0.0
0.0              0.0               0.0            0.0         0.0
0.0                         0.0          0.0           0.0           0.0
0.0                   0.0             0.0               0.0               0.0
0.0     0.0                 0.0       0.0     0.0                 0.0
0.0       0.0    0.0       0.0       0.0          0.0               0.0
0.0              0.0                   0.0          0.0             80.0
0.0              0.0                   0.0         0.0       0.0       0.0
0.0         0.0  0.0      0.0                   0.0         0.0            0.0
0.0      0.0          0.0             0.0                 0.0               0.0
0.0       0.0             0.0            0.0                           0.0
0.0              0.0                 0.0             0.0              0.0
0.0         0.0          0.0     0.0           0.0         0.0
0.0                         0.0      0.0                     0.0
0.0
28      00ba244566e36e0af3d979320fd3017f            0.0           0.0
0.0       0.0                               0.0          0.0                 0.0
0.0     0.0     0.0                 0.0                   0.0
13.0              0.0         0.0       0.0            0.0             0.0
```

```
0.0          0.0                    0.0    0.0                0.0     0.0
0.0           0.0    0.0       0.0                        0.0        0.0   0.0
0.0             0.0     0.0                     0.0            0.0    0.0
0.0       0.0               0.0             0.0               0.0          0.0
0.0                     0.0             0.0              0.0
0.0             0.0               0.0            0.0         0.0
0.0                         0.0          0.0           0.0           0.0
0.0                 0.0              0.0                0.0               0.0
0.0     0.0                 0.0       0.0     0.0                 0.0
0.0       0.0    0.0       0.0       0.0          0.0               0.0
0.0              0.0                  0.0           0.0             0.0
0.0             0.0                   0.0         0.0       0.0       0.0
0.0         0.0  0.0      0.0                  0.0         0.0            0.0
0.0      0.0          0.0              0.0                 0.0               0.0
0.0       0.0             0.0            0.0                           0.0
0.0             0.0                 0.0             0.0             0.0
0.0         0.0          0.0     0.0           0.0         0.0
0.0                          0.0      0.0                      0.0
0.0
37     0100f55e4f0fe28f2c0465d3fc4b9897            0.0           0.0
0.0       0.0                                0.0          0.0                 0.0
0.0     0.0     0.0                 0.0                   0.0
0.0               0.0         0.0       0.0            0.0             0.0
0.0          0.0                    0.0    0.0               0.0     0.0
0.0           0.0    0.0       0.0                        0.0        0.0   0.0
0.0             0.0     0.0                     0.0            0.0    0.0
0.0       0.0               0.0             0.0               0.0          0.0
0.0                     0.0             0.0              0.0
0.0             0.0              50.0            0.0         0.0
0.0                         0.0          0.0           0.0           0.0
0.0                 0.0              0.0                0.0               0.0
0.0     0.0                 0.0       0.0     0.0                 0.0
0.0       0.0    0.0       0.0       0.0          0.0               0.0
0.0              0.0                  0.0           0.0             0.0
0.0             0.0                   0.0         0.0       0.0       0.0
0.0         0.0  0.0      0.0                  0.0         0.0            0.0
0.0      0.0          0.0              0.0                 0.0               0.0
0.0       0.0             0.0            0.0                           0.0
0.0             0.0                 0.0             0.0             0.0
0.0         0.0          0.0     0.0           0.0         0.0
0.0                          0.0      0.0                      0.0
0.0
```

The images are split into train, validate, and test groups. Dog images are cropped and transformed into three channel inputs, which are further flattened for the sequential network with linear hidden layers.

[276]:
```
img_transform = {
    'valid':transforms.Compose([
        transforms.Resize(size = 256, interpolation = transforms.
↪InterpolationMode.NEAREST),
        transforms.CenterCrop(size = 224),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406],
                             [0.229, 0.224, 0.225])
    ]),
    'train':transforms.Compose([
       transforms.RandomResizedCrop(size = 256, interpolation = transforms.
↪InterpolationMode.NEAREST),
        transforms.RandomRotation(degrees = 30),
        transforms.ColorJitter(),
        transforms.RandomHorizontalFlip(),
        transforms.CenterCrop(size=224),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406],
                             [0.229, 0.224, 0.225])
    ]),
    'test':transforms.Compose([
        transforms.Resize(size = 256, interpolation = transforms.
↪InterpolationMode.NEAREST),
        transforms.CenterCrop(size = 224),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406],
                             [0.229, 0.224, 0.225])
    ]),
}
```

[277]:
```
class DogBreedDataset(torch.utils.data.Dataset):
    'Characterizes a dataset for PyTorch'
    def __init__(self, img_dir, label, transform):
        'Initialization'
        self.img_dir = img_dir
        self.transform = transform
        self.label = label

    def __len__(self):
        'Denotes the total number of samples'
        return len(self.label)

    def __getitem__(self, index):
        if self.label is not None:
            img_name = '{}.jpg'.format(self.label.iloc[index, 0])
            fullname = self.img_dir + img_name
            image = Image.open(fullname)
```

```
            label = self.label.iloc[index, 1:].astype('float').to_numpy()
            label = np.argmax(label)
            if self.transform:
                image = self.transform(image)
            return [image, label]
```

[278]:
```
batch_size = 12
num_workers = 4
train_img = DogBreedDataset('/content/dog-breed-identification/train/', train,␣
↪transform = img_transform['train'])
valid_img = DogBreedDataset('/content/dog-breed-identification/train/', valid,␣
↪transform = img_transform['valid'])


dataloaders={
    'train':torch.utils.data.DataLoader(train_img, batch_size, num_workers =␣
↪num_workers, shuffle=True),
    'valid':torch.utils.data.DataLoader(valid_img, batch_size, num_workers =␣
↪num_workers, shuffle=False)
}
```

[279]:
```
train_img.transform
```

[279]:
```
Compose(
    RandomResizedCrop(size=(256, 256), scale=(0.08, 1.0), ratio=(0.75, 1.3333),
interpolation=nearest)
    RandomRotation(degrees=[-30.0, 30.0], interpolation=nearest, expand=False,
fill=0)
    ColorJitter(brightness=None, contrast=None, saturation=None, hue=None)
    RandomHorizontalFlip(p=0.5)
    CenterCrop(size=(224, 224))
    ToTensor()
    Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
)
```

[280]:
```
use_cuda = torch.cuda.is_available()
```

[281]:
```
def imshow(axis, inp):
    """Denormalize and show"""
    inp = inp.numpy().transpose((1, 2, 0))
    mean = np.array([0.485, 0.456, 0.406])
    std = np.array([0.229, 0.224, 0.225])
    inp = std * inp + mean
    axis.imshow(inp)
```

[282]:
```
type(dataloaders['train'])
```

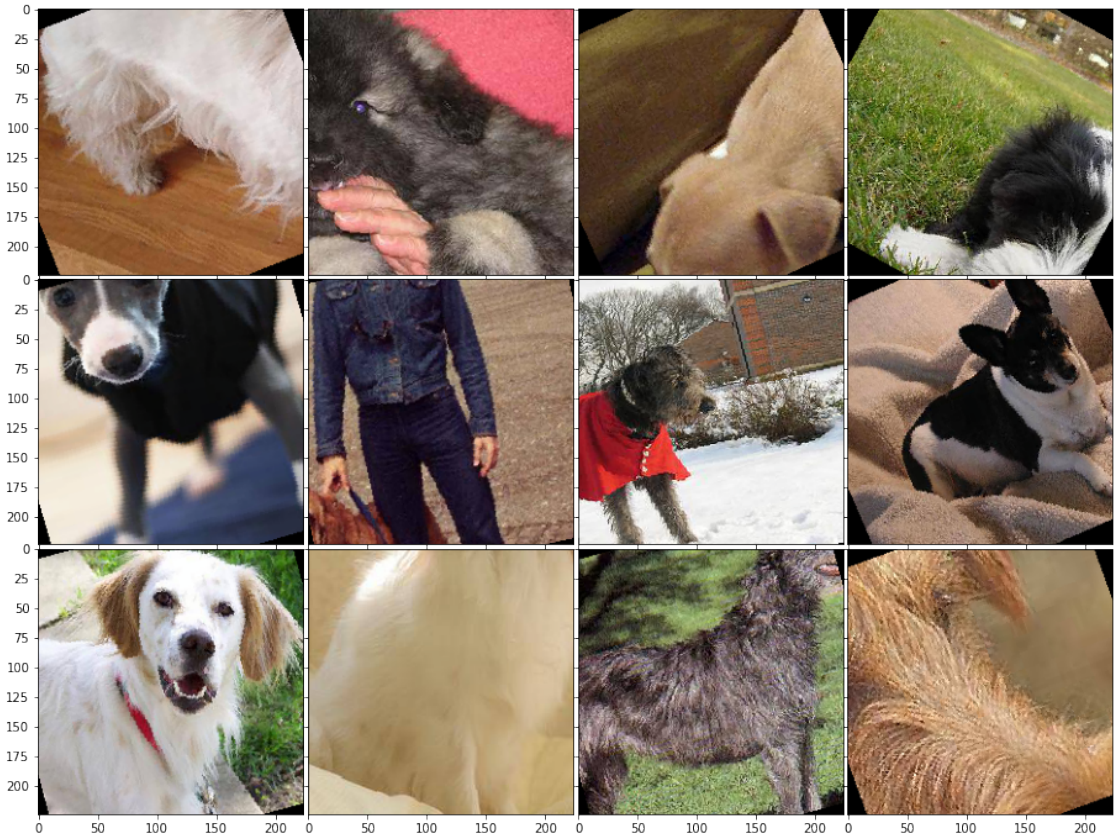
[282]:
```
torch.utils.data.dataloader.DataLoader
```

The preprocessed batch images are randomly adjusted and cropped to prevent overfitting and provide more data. Sometimes this results in the dog being unrecognizable by human eyes though.

[283]:
```python
from mpl_toolkits.axes_grid1 import ImageGrid
img, label = next(iter(dataloaders['train']))
print(img.size(), label.size())
fig = plt.figure(1, figsize=(16, 12))
grid = ImageGrid(fig, 111, nrows_ncols=(3, 4), axes_pad=0.05)
for i in range(img.size()[0]):
    ax = grid[i]
    imshow(ax, img[i])
```

```
torch.Size([12, 3, 224, 224]) torch.Size([12])
```

```
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers).
```



0.0.2 Sequential Model

[285]:
```
import torch.nn as nn

##  Specify model architecture
model_sequential = nn.Sequential(
    nn.LazyLinear(512),
    nn.ReLU(),
    nn.Dropout(p = 0.3),
    nn.LazyLinear(256),
    nn.ReLU(),
    nn.Dropout(p = 0.3),
    nn.LazyLinear(120)
)

if use_cuda:
    model_sequential = model_sequential.cuda()
```

Train is the driving function behind all of the models, and it handles the loss and activation functions for forward and back propagation. For each epoch, accuracy and loss are plotted, and

the model will then be saved to Google Drive for later evaluation because training takes hours for each model.

[286]:
```
def train(n_epochs, loaders, model, optimizer, criterion, use_cuda, save_path,␣
↪flatten):
    """returns trained model"""
    # initialize tracker for minimum validation loss
    valid_loss_min = np.Inf

    all_train_acc = []
    all_train_loss = []
    all_valid_acc = []
    all_valid_loss = []

    for epoch in range(1, n_epochs+1):
        # initialize variables to monitor training and validation loss
        train_loss = 0.0
        valid_loss = 0.0
        total_train = 0
        correct_train = 0
        total_valid = 0
        correct_valid = 0

        ###################
        # train the model #
        ###################
        model.train()
        for batch_idx, (data, target) in enumerate(loaders['train']):
            # move to GPU
            if use_cuda:
                data, target = data.cuda(), target.cuda()
            ## find the loss and update the model parameters accordingly
            ## record the average training loss, using something like

            optimizer.zero_grad()
            if flatten:
              flat_data = data.view(data.size(0), -1)
              output = model(flat_data)
            else:
              output = model(data)
            loss = criterion(output, target)
            loss.backward()
            optimizer.step()
            train_loss = train_loss + ((1 / (batch_idx + 1)) * (loss.data -␣
↪train_loss))

            _, predicted = output.max(1)
```

```
            total_train += target.size(0)
            correct_train += predicted.eq(target).sum().item()

            if batch_idx % 100 == 0:
                print('Epoch: %d \tBatch: %d \tTraining Loss: %.6f' %(epoch,␣
↪batch_idx + 1, train_loss))
        ######################
        # validate the model #
        ######################
        model.eval()
        for batch_idx, (data, target) in enumerate(loaders['valid']):
            # move to GPU
            if use_cuda:
                data, target = data.cuda(), target.cuda()
            ## update the average validation loss
            if flatten:
              flat_data = data.view(data.size(0), -1)
              output = model(flat_data)
            else:
              output = model(data)
            loss = criterion(output, target)
            valid_loss = valid_loss + ((1 / (batch_idx + 1)) * (loss.data -␣
↪valid_loss))

            _, predicted = output.max(1)
            total_valid += target.size(0)
            correct_valid += predicted.eq(target).sum().item()

        # print training/validation statistics
        print('Epoch: {} \tTraining Loss: {:.4f} \tValidation Loss: {:.4f}'.
↪format(
            epoch,
            train_loss,
            valid_loss
            ))

        train_acc = 100 * (correct_train / total_train)
        all_train_acc.append(train_acc)
        all_train_loss.append(train_loss)

        valid_acc= 100 * (correct_valid / total_valid)
        all_valid_acc.append(valid_acc)
        all_valid_loss.append(valid_loss)

        ## TODO: save the model if validation loss has decreased
        if valid_loss < valid_loss_min:
            torch.save(model.state_dict(), save_path)
```

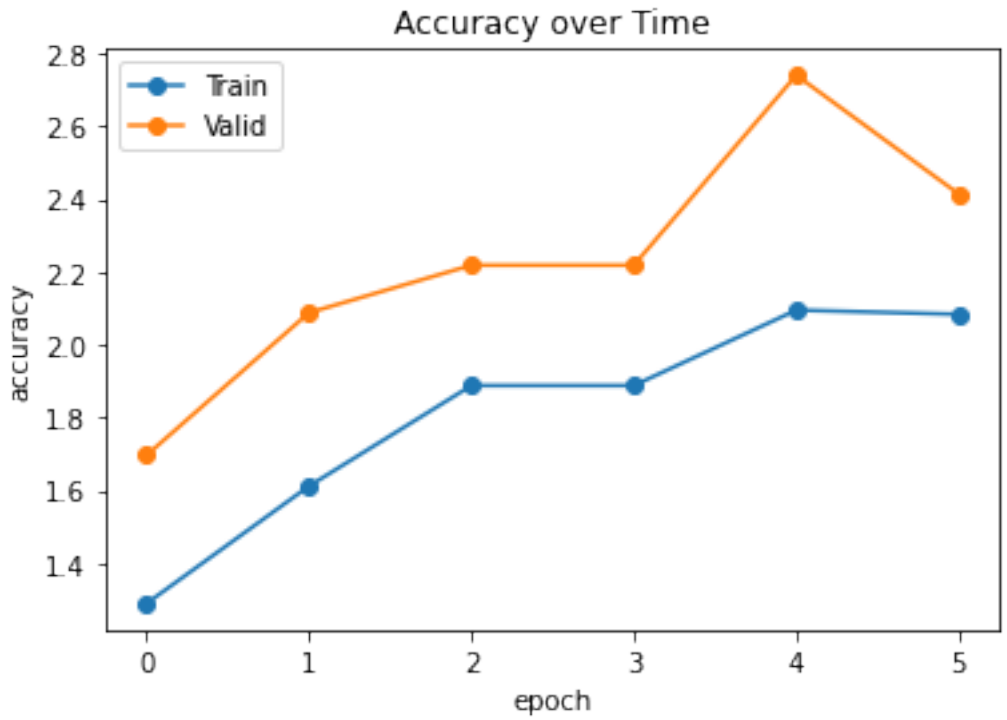
```
            print('BOOM! Validation loss decreased ({:.4f} --> {:.4f}).  Saving␣
↪model...'.format(valid_loss_min,valid_loss))
            valid_loss_min = valid_loss

    ## Plot accuracy and loss over epochs
    plt.plot(all_train_acc,'-o')
    plt.plot(all_valid_acc,'-o')
    plt.xlabel('epoch')
    plt.ylabel('accuracy')
    plt.legend(['Train','Valid'])
    plt.title('Accuracy over Time')

    plt.show()

    plt.plot(all_train_loss,'-o')
    plt.plot(all_valid_loss,'-o')
    plt.xlabel('epoch')
    plt.ylabel('losses')
    plt.legend(['Train','Valid'])
    plt.title('Loss over Time')

    plt.show()

    # return trained model
    return model
```

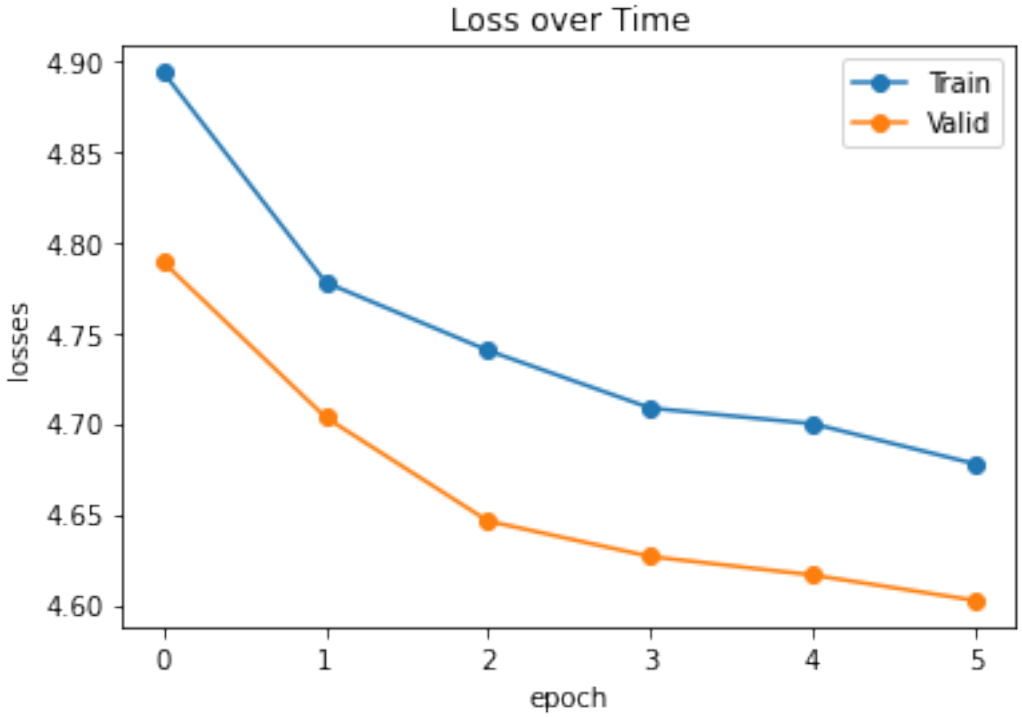
[287]:
```
criterion_seq = nn.CrossEntropyLoss()
optimizer_seq = torch.optim.SGD(model_sequential.parameters(), lr=0.005)␣
↪#activation
n_epochs = 6
```

[288]:
```
# train the model
model_sequential = train(n_epochs, dataloaders, model_sequential,␣
↪optimizer_seq, criterion_seq, use_cuda, 'model_sequential.pt', True)
```

```
Epoch: 1        Batch: 1        Training Loss: 4.853119
Epoch: 1        Batch: 101      Training Loss: 5.025080
Epoch: 1        Batch: 201      Training Loss: 4.992989
Epoch: 1        Batch: 301      Training Loss: 4.974000
Epoch: 1        Batch: 401      Training Loss: 4.951247
Epoch: 1        Batch: 501      Training Loss: 4.925786
Epoch: 1        Batch: 601      Training Loss: 4.910930
Epoch: 1        Batch: 701      Training Loss: 4.897637
Epoch: 1        Training Loss: 4.8936   Validation Loss: 4.7889
BOOM! Validation loss decreased (inf --> 4.7889).  Saving model…
Epoch: 2        Batch: 1        Training Loss: 4.781469
Epoch: 2        Batch: 101      Training Loss: 4.802196
```

```
Epoch: 2        Batch: 201      Training Loss: 4.786203
Epoch: 2        Batch: 301      Training Loss: 4.784838
Epoch: 2        Batch: 401      Training Loss: 4.784549
Epoch: 2        Batch: 501      Training Loss: 4.783294
Epoch: 2        Batch: 601      Training Loss: 4.783128
Epoch: 2        Batch: 701      Training Loss: 4.779267
Epoch: 2        Training Loss: 4.7777   Validation Loss: 4.7033
BOOM! Validation loss decreased (4.7889 --> 4.7033).  Saving model…
Epoch: 3        Batch: 1        Training Loss: 4.621927
Epoch: 3        Batch: 101      Training Loss: 4.755564
Epoch: 3        Batch: 201      Training Loss: 4.758684
Epoch: 3        Batch: 301      Training Loss: 4.747759
Epoch: 3        Batch: 401      Training Loss: 4.741254
Epoch: 3        Batch: 501      Training Loss: 4.741363
Epoch: 3        Batch: 601      Training Loss: 4.736922
Epoch: 3        Batch: 701      Training Loss: 4.740412
Epoch: 3        Training Loss: 4.7401   Validation Loss: 4.6461
BOOM! Validation loss decreased (4.7033 --> 4.6461).  Saving model…
Epoch: 4        Batch: 1        Training Loss: 4.725187
Epoch: 4        Batch: 101      Training Loss: 4.716299
Epoch: 4        Batch: 201      Training Loss: 4.723058
Epoch: 4        Batch: 301      Training Loss: 4.715782
Epoch: 4        Batch: 401      Training Loss: 4.717066
Epoch: 4        Batch: 501      Training Loss: 4.714409
Epoch: 4        Batch: 601      Training Loss: 4.711252
Epoch: 4        Batch: 701      Training Loss: 4.707241
Epoch: 4        Training Loss: 4.7084   Validation Loss: 4.6265
BOOM! Validation loss decreased (4.6461 --> 4.6265).  Saving model…
Epoch: 5        Batch: 1        Training Loss: 4.742851
Epoch: 5        Batch: 101      Training Loss: 4.676271
Epoch: 5        Batch: 201      Training Loss: 4.693408
Epoch: 5        Batch: 301      Training Loss: 4.701306
Epoch: 5        Batch: 401      Training Loss: 4.694970
Epoch: 5        Batch: 501      Training Loss: 4.693490
Epoch: 5        Batch: 601      Training Loss: 4.695495
Epoch: 5        Batch: 701      Training Loss: 4.697802
Epoch: 5        Training Loss: 4.6997   Validation Loss: 4.6163
BOOM! Validation loss decreased (4.6265 --> 4.6163).  Saving model…
Epoch: 6        Batch: 1        Training Loss: 4.695712
Epoch: 6        Batch: 101      Training Loss: 4.662190
Epoch: 6        Batch: 201      Training Loss: 4.676803
Epoch: 6        Batch: 301      Training Loss: 4.681264
Epoch: 6        Batch: 401      Training Loss: 4.685544
Epoch: 6        Batch: 501      Training Loss: 4.680231
Epoch: 6        Batch: 601      Training Loss: 4.679135
Epoch: 6        Batch: 701      Training Loss: 4.677979
Epoch: 6        Training Loss: 4.6776   Validation Loss: 4.6023
BOOM! Validation loss decreased (4.6163 --> 4.6023).  Saving model…
```


Accuracy over Time
Train
Valid
accuracy
epoch
2.8
2.6
2.4
2.2
2.0
1.8
1.6
1.4
0
1
2
3
4
5


Loss over Time
Train
Valid
losses
epoch
4.90
4.85
4.80
4.75
4.70
4.65
4.60
0
1
2
3
4
5

[289]:
```
import pickle
from os.path import exists
pickleFilname = f'/content/drive/MyDrive/College Junior/CS 4375/
↪model_sequentialAllImages{n_epochs}epochs.pkl'

if( exists(pickleFilname ) ):
    with open(pickleFilname , 'rb') as f:
        model_sequential = pickle.load(f)

with open(pickleFilname , 'wb') as f:
    pickle.dump(model_sequential, f)
```

[290]:
```
# model_transfer
from PIL import Image

if (False): #this is a testing cell
  #filePath = '/content/dog-breed-identification/train/
↪0a3f1898556115d6d0931294876cd1d9.jpg' = maltese dog   3 epoch, 1000 images
  #filePath = '/content/dog-breed-identification/train/
↪0b97116ed04c8f0f7eb4a2b4b2620476.jpg' = samoyed (ish) 3 epoch, 1000 images
  filePath = '/content/dog-breed-identification/train/
↪0d103ca7cf575757374f8f6ae87d8868.jpg' #= miniature_pinscher 3 epoch, 10222␣
↪images
  img_dog = Image.open(filePath).convert('RGB')
  from torchvision import transforms

  # Create a preprocessing pipeline
  preprocess = transforms.Compose([
          transforms.Resize(256),
          transforms.CenterCrop(224),
          transforms.ToTensor(),
          transforms.Normalize(
          mean=[0.485, 0.456, 0.406],
          std=[0.229, 0.224, 0.225]
      )])

  # Pass the image for preprocessing and the image preprocessed
  # Reshape, crop, and normalize the input tensor for feeding into network for␣
↪evaluation
  img_dog_preprocessed = preprocess(img_dog)
  batch_img_dog_tensor = torch.unsqueeze(img_dog_preprocessed, 0)

  prediction = model_sequential.forward(batch_img_dog_tensor)
  print(prediction)

  list(prediction)[0]
  list(list(prediction)[0])
```

```
  prediction.shape
```

[291]:
```
if(False): #this is a testing cell
  model_sequential.eval()
  out = model_sequential(batch_img_dog_tensor)
  _, index = torch.max(out, 1)

  print("index:", index)
  percentage = torch.nn.functional.softmax(out, dim=1)[0] * 100
  print()
  print(percentage)
  newLabels = list(labels['target'])
  # Print the name along with score of the object identified by the model

  print(newLabels[index[0]], percentage[index[0]].item())

  # Print the top 5 scores along with the image label. Sort function is invoked␣
 ↪on the torch to sort the scores.

  _, indices = torch.sort(out, descending=True)
  [(newLabels[idx], percentage[idx].item()) for idx in indices[0][:5]]
  print("predicted label:", labelnames.columns[index+1])
```

This function handles label identification and printing for each dog image, and it is used for evaluation on test data.

[292]:
```
def identifyDog(file, df, bShow = False):
  img_dog = Image.open(file).convert('RGB')
  #preprocessing pipeline
  preprocess = transforms.Compose([
          transforms.Resize(256),
          transforms.CenterCrop(224),
          transforms.ToTensor(),
          transforms.Normalize(
          mean=[0.485, 0.456, 0.406],
          std=[0.229, 0.224, 0.225]
      )])
  img_dog_preprocessed = preprocess(img_dog)
  batch_img_dog_tensor = torch.unsqueeze(img_dog_preprocessed, 0)

  #create prediction array
  prediction = model_transfer.forward(batch_img_dog_tensor)
  list(prediction)[0]
  list(list(prediction)[0])

  #pick out the most likely breed from predictions
  model_transfer.eval()
```

```
 out = model_transfer(batch_img_dog_tensor)
 _, index = torch.max(out, 1)

 #choose the label that matches the index
 percentage = torch.nn.functional.softmax(out, dim=1)[0] * 100
 newLabels = list(labels['target'])

 #find actual breed in label csv
 theId = file.split('/')[-1].replace('.jpg','')
 theBreed = list(labels[labels['id'].str.contains(theId)]['breed'])[0]

 #print prediction and actual breed
 predBreed = labelnames.columns[index+1]
 #print("Predicted dog breed:", predBreed)
 #print("Actual dog breed:", theBreed)

 dict = { "Actual Label": theBreed, "Predicted Label": predBreed, "Match":␣
↪"True" if theBreed == predBreed else "False"}
 df = df.append(dict, ignore_index = True)

 if(bShow):
 #plot dog image
   img = cv2.imread(file)
   img_cvt=cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
   plt.imshow(img_cvt)
   plt.show()

 bSuccess = True

 #if prediction is wrong, plot the predicted dog
 if(theBreed != predBreed):
   bSuccess = False
   if(bShow):
     #print(f"The predicted breed is wrong. Here is what a {predBreed}␣
↪normally looks like.")
     #find what the predicted breed is
     thePredBreedId = list(labels[labels['breed'].str.
↪contains(predBreed)]['id'])[0]
     #plot that dog
     file = f"/content/dog-breed-identification/train/{thePredBreedId}.jpg"
     img = cv2.imread(file)
     img_cvt=cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
     plt.imshow(img_cvt)
     plt.show()

 return df, bSuccess
```

[293]:
```
df = pd.DataFrame()
numCorrects = 0
numWrongs = 0

numImages = 100
for id in listOfTest[0:numImages]:
  filePath = f"/content/dog-breed-identification/train/{id}.jpg"
  df, bSuccess = identifyDog(filePath, df)
  if(bSuccess):
    numCorrects += 1
  else:
    numWrongs += 1

accuracy = numCorrects/(numCorrects+numWrongs)
print("Number of correct predictions: ", numCorrects)
print("Number of incorrect predictions: ", numWrongs)
print(f"Accuracy of the model for {numImages} dogs: ","{:.2f}%".
 ↪format(accuracy*100))
print(df.to_string())
```

```
Number of correct predictions:  2
Number of incorrect predictions:  98
Accuracy of the model for 100 dogs:  2.00%
                   Actual Label             Predicted Label  Match
0            bedlington_terrier          australian_terrier  False
1               norfolk_terrier          australian_terrier  False
2            standard_schnauzer          australian_terrier  False
3       black-and-tan_coonhound          australian_terrier  False
4              golden_retriever          australian_terrier  False
5              lakeland_terrier          australian_terrier  False
6           african_hunting_dog          australian_terrier  False
7                 affenpinscher          australian_terrier  False
8    greater_swiss_mountain_dog          australian_terrier  False
9                  irish_setter          australian_terrier  False
10                   bloodhound          australian_terrier  False
11               english_setter          australian_terrier  False
12                     malamute        norwegian_elkhound  False
13                       saluki          australian_terrier  False
14                   weimaraner          australian_terrier  False
15           standard_schnauzer                      saluki  False
16              giant_schnauzer                      saluki  False
17                irish_terrier          australian_terrier  False
18                  entlebucher                      saluki  False
19          rhodesian_ridgeback                      saluki  False
20                        lhasa          australian_terrier  False
21                     pekinese             tibetan_terrier  False
22            italian_greyhound        norwegian_elkhound  False
```

23	lakeland_terrier	australian_terrier	False
24	border_terrier	saluki	False
25	lhasa	australian_terrier	False
26	leonberg	australian_terrier	False
27	bedlington_terrier	german_short-haired_pointer	False
28	eskimo_dog	australian_terrier	False
29	otterhound	australian_terrier	False
30	groenendael	australian_terrier	False
31	toy_poodle	tibetan_terrier	False
32	maltese_dog	german_short-haired_pointer	False
33	bernese_mountain_dog	saluki	False
34	chow	australian_terrier	False
35	affenpinscher	australian_terrier	False
36	basenji	australian_terrier	False
37	bloodhound	australian_terrier	False
38	bluetick	australian_terrier	False
39	welsh_springer_spaniel	tibetan_terrier	False
40	old_english_sheepdog	australian_terrier	False
41	scottish_deerhound	australian_terrier	False
42	maltese_dog	australian_terrier	False
43	afghan_hound	australian_terrier	False
44	kerry_blue_terrier	german_short-haired_pointer	False
45	saluki	saluki	True
46	african_hunting_dog	australian_terrier	False
47	norfolk_terrier	australian_terrier	False
48	shih-tzu	australian_terrier	False
49	great_pyrenees	australian_terrier	False
50	toy_poodle	german_short-haired_pointer	False
51	toy_poodle	australian_terrier	False
52	leonberg	australian_terrier	False
53	clumber	australian_terrier	False
54	irish_terrier	australian_terrier	False
55	japanese_spaniel	german_short-haired_pointer	False
56	basset	australian_terrier	False
57	norwegian_elkhound	norwegian_elkhound	True
58	great_dane	australian_terrier	False
59	blenheim_spaniel	german_short-haired_pointer	False
60	chow	norwegian_elkhound	False
61	dhole	norwegian_elkhound	False
62	miniature_schnauzer	saluki	False
63	german_short-haired_pointer	australian_terrier	False
64	airedale	australian_terrier	False
65	komondor	australian_terrier	False
66	newfoundland	australian_terrier	False
67	west_highland_white_terrier	australian_terrier	False
68	pomeranian	norwegian_elkhound	False
69	german_shepherd	australian_terrier	False
70	welsh_springer_spaniel	australian_terrier	False

```
71                 appenzeller                      saluki  False
72              cocker_spaniel             tibetan_terrier  False
73          scottish_deerhound          australian_terrier  False
74             irish_wolfhound          australian_terrier  False
75                      briard          australian_terrier  False
76                       chow          australian_terrier  False
77                      kuvasz          australian_terrier  False
78                     basenji                      saluki  False
79           english_springer  german_short-haired_pointer  False
80                      vizsla          australian_terrier  False
81           lakeland_terrier          australian_terrier  False
82           blenheim_spaniel          australian_terrier  False
83          brabancon_griffon          australian_terrier  False
84           golden_retriever  german_short-haired_pointer  False
85         kerry_blue_terrier  german_short-haired_pointer  False
86                      briard          australian_terrier  False
87                       boxer          australian_terrier  False
88            norfolk_terrier          australian_terrier  False
89     welsh_springer_spaniel  german_short-haired_pointer  False
90                  chihuahua          australian_terrier  False
91          norwegian_elkhound          australian_terrier  False
92          bedlington_terrier  german_short-haired_pointer  False
93  greater_swiss_mountain_dog                      saluki  False
94                   cardigan                      saluki  False
95                    redbone  german_short-haired_pointer  False
96              affenpinscher          australian_terrier  False
97                       dhole          australian_terrier  False
98                boston_bull          australian_terrier  False
99            norwich_terrier          australian_terrier  False
```

0.0.3 CNN Without Transfer Learning

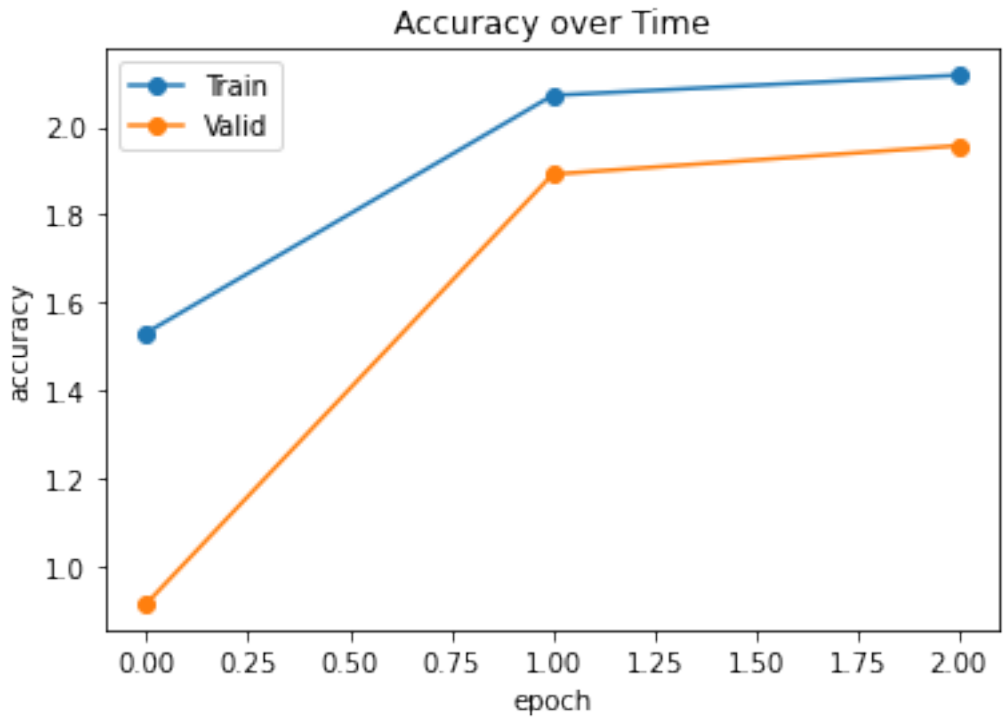
[262]:
```
import torch.nn as nn

##  Specify model architecture
model_cnn = nn.Sequential(
    nn.Conv2d(3, 32, 3, padding = 1),
    nn.ReLU(),
    nn.Conv2d(32, 32, 3, padding = 1),
    nn.ReLU(),
    nn.MaxPool2d(2),
    nn.Conv2d(32, 64, 3, padding = 1),
    nn.ReLU(),
    nn.Conv2d(64, 64, 3, padding = 1),
    nn.ReLU(),
    nn.MaxPool2d(2),
```

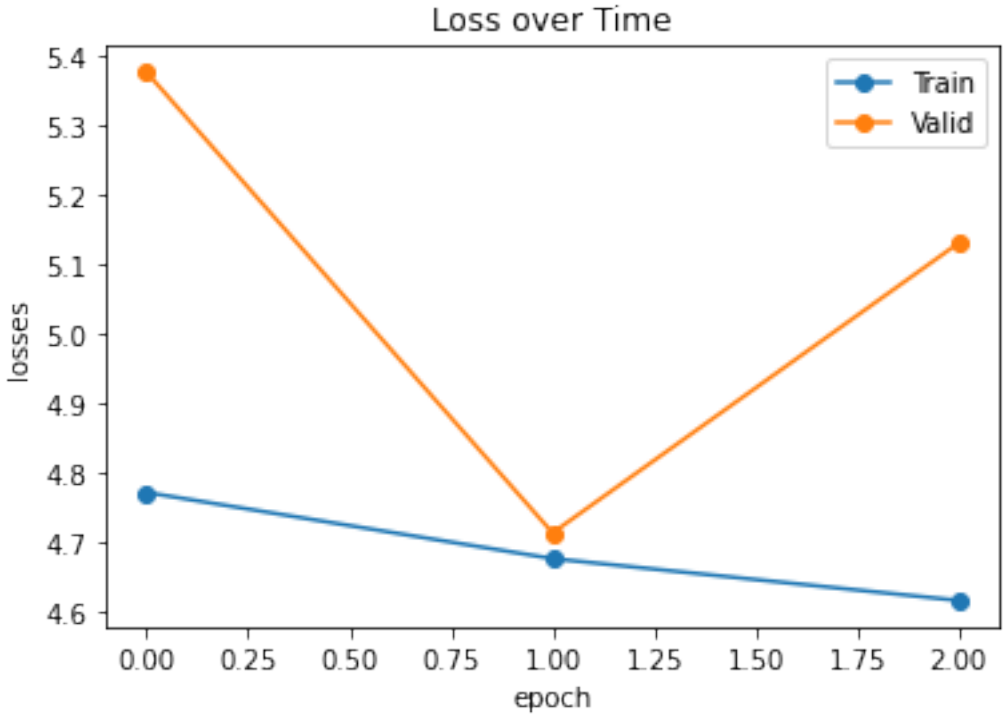
```
    nn.Flatten(),
    nn.LazyLinear(256),
    nn.ReLU(),
    nn.LazyLinear(120)
)

if use_cuda:
    model_sequential = model_sequential.cuda()
```

[263]:
```
criterion_cnn = nn.CrossEntropyLoss()
optimizer_cnn = torch.optim.SGD(model_cnn.parameters(), lr=0.05) #activation
n_epochs = 3
```

[264]:
```
# train the model
model_cnn = train(n_epochs, dataloaders, model_cnn, optimizer_cnn,␣
 ↪criterion_cnn, use_cuda, 'model_cnn.pt', False)
```

```
Epoch: 1        Batch: 1        Training Loss: 4.804548
Epoch: 1        Batch: 101      Training Loss: 4.791172
Epoch: 1        Batch: 201      Training Loss: 4.789042
Epoch: 1        Batch: 301      Training Loss: 4.788400
Epoch: 1        Batch: 401      Training Loss: 4.786088
Epoch: 1        Batch: 501      Training Loss: 4.781695
Epoch: 1        Batch: 601      Training Loss: 4.779020
Epoch: 1        Batch: 701      Training Loss: 4.774522
Epoch: 1        Training Loss: 4.7725   Validation Loss: 5.3759
BOOM! Validation loss decreased (inf --> 5.3759).  Saving model…
Epoch: 2        Batch: 1        Training Loss: 5.691835
Epoch: 2        Batch: 101      Training Loss: 4.708488
Epoch: 2        Batch: 201      Training Loss: 4.703941
Epoch: 2        Batch: 301      Training Loss: 4.695369
Epoch: 2        Batch: 401      Training Loss: 4.688232
Epoch: 2        Batch: 501      Training Loss: 4.688097
Epoch: 2        Batch: 601      Training Loss: 4.685279
Epoch: 2        Batch: 701      Training Loss: 4.679914
Epoch: 2        Training Loss: 4.6776   Validation Loss: 4.7145
BOOM! Validation loss decreased (5.3759 --> 4.7145).  Saving model…
Epoch: 3        Batch: 1        Training Loss: 4.821936
Epoch: 3        Batch: 101      Training Loss: 4.681162
Epoch: 3        Batch: 201      Training Loss: 4.655678
Epoch: 3        Batch: 301      Training Loss: 4.640582
Epoch: 3        Batch: 401      Training Loss: 4.637477
Epoch: 3        Batch: 501      Training Loss: 4.629827
Epoch: 3        Batch: 601      Training Loss: 4.621590
Epoch: 3        Batch: 701      Training Loss: 4.617757
Epoch: 3        Training Loss: 4.6175   Validation Loss: 5.1307
```


Accuracy over Time
Train
Valid
accuracy
epoch
1.0
1.2
1.4
1.6
1.8
2.0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00


Loss over Time
Train
Valid
losses
epoch
4.6
4.7
4.8
4.9
5.0
5.1
5.2
5.3
5.4
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00

[265]:
```
import pickle
from os.path import exists
pickleFilname = f'/content/drive/MyDrive/College Junior/CS 4375/
↪model_cnnAllImages{n_epochs}epochs.pkl'

if( exists(pickleFilname ) ):
    with open(pickleFilname , 'rb') as f:
        model_cnn = pickle.load(f)

with open(pickleFilname , 'wb') as f:
    pickle.dump(model_cnn, f)
```

[266]:
```
# model_transfer
from PIL import Image

if (False): #this is a testing cell
  #filePath = '/content/dog-breed-identification/train/
↪0a3f1898556115d6d0931294876cd1d9.jpg' = maltese dog   3 epoch, 1000 images
  #filePath = '/content/dog-breed-identification/train/
↪0b97116ed04c8f0f7eb4a2b4b2620476.jpg' = samoyed (ish) 3 epoch, 1000 images
  filePath = '/content/dog-breed-identification/train/
↪0d103ca7cf575757374f8f6ae87d8868.jpg' #= miniature_pinscher 3 epoch, 10222␣
↪images
  img_dog = Image.open(filePath).convert('RGB')
  from torchvision import transforms

  # Create a preprocessing pipeline
  preprocess = transforms.Compose([
          transforms.Resize(256),
          transforms.CenterCrop(224),
          transforms.ToTensor(),
          transforms.Normalize(
          mean=[0.485, 0.456, 0.406],
          std=[0.229, 0.224, 0.225]
      )])

  # Pass the image for preprocessing and the image preprocessed
  # Reshape, crop, and normalize the input tensor for feeding into network for␣
↪evaluation
  img_dog_preprocessed = preprocess(img_dog)
  batch_img_dog_tensor = torch.unsqueeze(img_dog_preprocessed, 0)

  prediction = model_cnn.forward(batch_img_dog_tensor)
  print(prediction)

  list(prediction)[0]
  list(list(prediction)[0])
```

```
  prediction.shape
```

[267]:
```
if(False): #this is a testing cell
  model_cnn.eval()
  out = model_cnn(batch_img_dog_tensor)
  _, index = torch.max(out, 1)

  print("index:", index)
  percentage = torch.nn.functional.softmax(out, dim=1)[0] * 100
  print()
  print(percentage)
  newLabels = list(labels['target'])
  # Print the name along with score of the object identified by the model

  print(newLabels[index[0]], percentage[index[0]].item())

  # Print the top 5 scores along with the image label. Sort function is invoked␣
 ↪on the torch to sort the scores.

  _, indices = torch.sort(out, descending=True)
  [(newLabels[idx], percentage[idx].item()) for idx in indices[0][:5]]
  print("predicted label:", labelnames.columns[index+1])
```

[268]:
```
df = pd.DataFrame()
numCorrects = 0
numWrongs = 0

numImages = 100
for id in listOfTest[0:numImages]:
  filePath = f"/content/dog-breed-identification/train/{id}.jpg"
  df, bSuccess = identifyDog(filePath, df)
  if(bSuccess):
    numCorrects += 1
  else:
    numWrongs += 1

accuracy = numCorrects/(numCorrects+numWrongs)
print("Number of correct predictions: ", numCorrects)
print("Number of incorrect predictions: ", numWrongs)
print(f"Accuracy of the model for {numImages} dogs: ","{:.2f}%".
 ↪format(accuracy*100))
print(df.to_string())
```

```
Number of correct predictions:  0
Number of incorrect predictions:  100
Accuracy of the model for 100 dogs:  0.00%
                      Actual Label              Predicted Label  Match
```

0	dingo	saluki	False
1	pekinese	australian_terrier	False
2	golden_retriever	australian_terrier	False
3	scottish_deerhound	australian_terrier	False
4	golden_retriever	german_short-haired_pointer	False
5	giant_schnauzer	australian_terrier	False
6	golden_retriever	australian_terrier	False
7	irish_water_spaniel	german_short-haired_pointer	False
8	affenpinscher	australian_terrier	False
9	malamute	norwegian_elkhound	False
10	weimaraner	australian_terrier	False
11	standard_schnauzer	saluki	False
12	leonberg	australian_terrier	False
13	irish_terrier	australian_terrier	False
14	scotch_terrier	australian_terrier	False
15	border_terrier	saluki	False
16	lhasa	australian_terrier	False
17	afghan_hound	tibetan_terrier	False
18	borzoi	saluki	False
19	chow	australian_terrier	False
20	affenpinscher	australian_terrier	False
21	american_staffordshire_terrier	australian_terrier	False
22	english_setter	australian_terrier	False
23	african_hunting_dog	australian_terrier	False
24	english_foxhound	australian_terrier	False
25	old_english_sheepdog	australian_terrier	False
26	airedale	australian_terrier	False
27	black-and-tan_coonhound	australian_terrier	False
28	bedlington_terrier	german_short-haired_pointer	False
29	eskimo_dog	australian_terrier	False
30	west_highland_white_terrier	australian_terrier	False
31	irish_setter	german_short-haired_pointer	False
32	keeshond	norwegian_elkhound	False
33	pug	australian_terrier	False
34	japanese_spaniel	australian_terrier	False
35	basset	australian_terrier	False
36	whippet	saluki	False
37	pomeranian	norwegian_elkhound	False
38	irish_water_spaniel	german_short-haired_pointer	False
39	italian_greyhound	saluki	False
40	boston_bull	australian_terrier	False
41	pembroke	norwegian_elkhound	False
42	pomeranian	basenji	False
43	japanese_spaniel	german_short-haired_pointer	False
44	yorkshire_terrier	australian_terrier	False
45	pekinese	australian_terrier	False
46	collie	australian_terrier	False
47	basset	australian_terrier	False

48	bernese_mountain_dog	old_english_sheepdog	False
49	cairn	saluki	False
50	chihuahua	australian_terrier	False
51	whippet	saluki	False
52	basenji	saluki	False
53	komondor	australian_terrier	False
54	otterhound	australian_terrier	False
55	kuvasz	saluki	False
56	redbone	german_short-haired_pointer	False
57	newfoundland	australian_terrier	False
58	basset	australian_terrier	False
59	samoyed	australian_terrier	False
60	collie	german_short-haired_pointer	False
61	irish_terrier	australian_terrier	False
62	samoyed	australian_terrier	False
63	collie	australian_terrier	False
64	appenzeller	saluki	False
65	bloodhound	australian_terrier	False
66	malamute	australian_terrier	False
67	border_collie	australian_terrier	False
68	kuvasz	australian_terrier	False
69	basenji	saluki	False
70	kuvasz	australian_terrier	False
71	border_terrier	australian_terrier	False
72	chesapeake_bay_retriever	australian_terrier	False
73	newfoundland	australian_terrier	False
74	mexican_hairless	australian_terrier	False
75	otterhound	australian_terrier	False
76	yorkshire_terrier	australian_terrier	False
77	samoyed	australian_terrier	False
78	dandie_dinmont	australian_terrier	False
79	border_terrier	australian_terrier	False
80	entlebucher	german_short-haired_pointer	False
81	dingo	saluki	False
82	flat-coated_retriever	german_short-haired_pointer	False
83	cairn	australian_terrier	False
84	norfolk_terrier	australian_terrier	False
85	siberian_husky	norwegian_elkhound	False
86	gordon_setter	norwegian_elkhound	False
87	doberman	australian_terrier	False
88	bloodhound	australian_terrier	False
89	toy_poodle	saluki	False
90	english_foxhound	saluki	False
91	border_terrier	australian_terrier	False
92	welsh_springer_spaniel	german_short-haired_pointer	False
93	welsh_springer_spaniel	german_short-haired_pointer	False
94	entlebucher	saluki	False
95	silky_terrier	australian_terrier	False

```
96                        cardigan                         saluki  False
97              rhodesian_ridgeback  german_short-haired_pointer  False
98                         pembroke           australian_terrier  False
99               kerry_blue_terrier           australian_terrier  False
```

0.0.4 Transfer Learning and Parameter Tuning

[53]:
```
##  Specify model architecture
model_transfer = models.resnet50(pretrained=True)

# Freeze training for all "features" layers
for param in model_transfer.parameters():
  param.requires_grad = False

# replace the last fully connected layer with a Linnear layer 133 output
in_features = model_transfer.fc.in_features     # in_features = 2048
model_transfer.fc = nn.Linear(in_features, 120) # Linear(in_features=2048,␣
 ↪out_features=120, bias=True)

if use_cuda:
    model_transfer = model_transfer.cuda()
```

```
Downloading: "https://download.pytorch.org/models/resnet50-0676ba61.pth" to
/root/.cache/torch/hub/checkpoints/resnet50-0676ba61.pth
```

```
  0%|          | 0.00/97.8M [00:00<?, ?B/s]
```

[54]:
```
criterion_transfer = nn.CrossEntropyLoss()
model_transfer_grad_paramaters = filter(lambda p: p.requires_grad,␣
 ↪model_transfer.parameters())
optimizer_transfer = torch.optim.SGD(model_transfer_grad_paramaters, lr=0.01)␣
 ↪#activation
n_epochs = 4
```

[ ]:
```
# train the model
model_transfer = train(n_epochs, dataloaders, model_transfer,␣
 ↪optimizer_transfer, criterion_transfer, use_cuda, 'model_transfer.pt', False)
```

[ ]:
```
f""" 1000 pictures, 3 epochs:
Epoch: 1         Batch: 1         Training Loss: 4.820606
Epoch: 1         Training Loss: 4.7854         Validation Loss: 4.6281
BOOM! Validation loss decreased (inf --> 4.6281).  Saving model...
Epoch: 2         Batch: 1         Training Loss: 4.775223
Epoch: 2         Training Loss: 4.4801         Validation Loss: 4.3470
BOOM! Validation loss decreased (4.6281 --> 4.3470).  Saving model...
Epoch: 3         Batch: 1         Training Loss: 4.042047
```

```
Epoch: 3         Training Loss: 4.2297         Validation Loss: 4.0708
BOOM! Validation loss decreased (4.3470 --> 4.0708).  Saving model...
"""

f2=""" 5000 pictures, 3 epochs:
Epoch: 1         Batch: 1         Training Loss: 4.832945
Epoch: 1         Batch: 101         Training Loss: 4.741760
Epoch: 1         Batch: 201         Training Loss: 4.606566
Epoch: 1         Batch: 301         Training Loss: 4.474421
Epoch: 1         Training Loss: 4.4165         Validation Loss: 3.4888
BOOM! Validation loss decreased (inf --> 3.4888).  Saving model...
Epoch: 2         Batch: 1         Training Loss: 3.961325
Epoch: 2         Batch: 101         Training Loss: 3.838313
Epoch: 2         Batch: 201         Training Loss: 3.725471
Epoch: 2         Batch: 301         Training Loss: 3.628852
Epoch: 2         Training Loss: 3.5870         Validation Loss: 2.5023
BOOM! Validation loss decreased (3.4888 --> 2.5023).  Saving model...
Epoch: 3         Batch: 1         Training Loss: 3.083075
Epoch: 3         Batch: 101         Training Loss: 3.201915
Epoch: 3         Batch: 201         Training Loss: 3.128031
Epoch: 3         Batch: 301         Training Loss: 3.071620
Epoch: 3         Training Loss: 3.0465         Validation Loss: 1.9330
BOOM! Validation loss decreased (2.5023 --> 1.9330).  Saving model..."""

f3=""" All 10222 pictures, 3 epochs:
Epoch: 1         Batch: 1         Training Loss: 4.947224
Epoch: 1         Batch: 101         Training Loss: 4.723919
Epoch: 1         Batch: 201         Training Loss: 4.587653
Epoch: 1         Batch: 301         Training Loss: 4.468295
Epoch: 1         Batch: 401         Training Loss: 4.346153
Epoch: 1         Batch: 501         Training Loss: 4.230396
Epoch: 1         Batch: 601         Training Loss: 4.116463
Epoch: 1         Batch: 701         Training Loss: 4.006574
Epoch: 1         Training Loss: 3.9866         Validation Loss: 2.7904
BOOM! Validation loss decreased (inf --> 2.7904).  Saving model...
Epoch: 2         Batch: 1         Training Loss: 3.871949
Epoch: 2         Batch: 101         Training Loss: 3.147888
Epoch: 2         Batch: 201         Training Loss: 3.082691
Epoch: 2         Batch: 301         Training Loss: 3.044378
Epoch: 2         Batch: 401         Training Loss: 2.997859
Epoch: 2         Batch: 501         Training Loss: 2.940052
Epoch: 2         Batch: 601         Training Loss: 2.892384
Epoch: 2         Batch: 701         Training Loss: 2.848122
Epoch: 2         Training Loss: 2.8419         Validation Loss: 1.6031
BOOM! Validation loss decreased (2.7904 --> 1.6031).  Saving model...
Epoch: 3         Batch: 1         Training Loss: 2.792788
Epoch: 3         Batch: 101         Training Loss: 2.471815
```

```
Epoch: 3         Batch: 201         Training Loss: 2.474647
Epoch: 3         Batch: 301         Training Loss: 2.458736
Epoch: 3         Batch: 401         Training Loss: 2.411387
Epoch: 3         Batch: 501         Training Loss: 2.375977
Epoch: 3         Batch: 601         Training Loss: 2.349866
Epoch: 3         Batch: 701         Training Loss: 2.331016
Epoch: 3         Training Loss: 2.3327         Validation Loss: 1.2304
BOOM! Validation loss decreased (1.6031 --> 1.2304).  Saving model...
"""

f4 = """ All 10222 pictures, 5 epochs
Epoch: 1         Batch: 1         Training Loss: 4.886767
Epoch: 1         Batch: 101         Training Loss: 4.729091
Epoch: 1         Batch: 201         Training Loss: 4.608284
Epoch: 1         Batch: 301         Training Loss: 4.482468
Epoch: 1         Batch: 401         Training Loss: 4.352062
Epoch: 1         Batch: 501         Training Loss: 4.239295
Epoch: 1         Batch: 601         Training Loss: 4.125138
Epoch: 1         Batch: 701         Training Loss: 4.019838
Epoch: 1         Training Loss: 3.9988         Validation Loss: 2.6609
BOOM! Validation loss decreased (inf --> 2.6609).  Saving model...
Epoch: 2         Batch: 1         Training Loss: 3.521045
Epoch: 2         Batch: 101         Training Loss: 3.167631
Epoch: 2         Batch: 201         Training Loss: 3.127581
Epoch: 2         Batch: 301         Training Loss: 3.069848
Epoch: 2         Batch: 401         Training Loss: 3.006356
Epoch: 2         Batch: 501         Training Loss: 2.958204
Epoch: 2         Batch: 601         Training Loss: 2.906377
Epoch: 2         Batch: 701         Training Loss: 2.867462
Epoch: 2         Training Loss: 2.8605         Validation Loss: 1.6696
BOOM! Validation loss decreased (2.6609 --> 1.6696).  Saving model...
Epoch: 3         Batch: 1         Training Loss: 3.659420
Epoch: 3         Batch: 101         Training Loss: 2.491639
Epoch: 3         Batch: 201         Training Loss: 2.491406
Epoch: 3         Batch: 301         Training Loss: 2.451811
Epoch: 3         Batch: 401         Training Loss: 2.420945
Epoch: 3         Batch: 501         Training Loss: 2.407779
Epoch: 3         Batch: 601         Training Loss: 2.376457
Epoch: 3         Batch: 701         Training Loss: 2.362266
Epoch: 3         Training Loss: 2.3613         Validation Loss: 1.1551
BOOM! Validation loss decreased (1.6696 --> 1.1551).  Saving model...
Epoch: 4         Batch: 1         Training Loss: 3.070820
Epoch: 4         Batch: 101         Training Loss: 2.260064
Epoch: 4         Batch: 201         Training Loss: 2.200973
Epoch: 4         Batch: 301         Training Loss: 2.158476
Epoch: 4         Batch: 401         Training Loss: 2.162050
Epoch: 4         Batch: 501         Training Loss: 2.149412
```

```
Epoch: 4         Batch: 601         Training Loss: 2.124513
Epoch: 4         Batch: 701         Training Loss: 2.121198
Epoch: 4         Training Loss: 2.1222         Validation Loss: 1.0137
BOOM! Validation loss decreased (1.1551 --> 1.0137).  Saving model...
Epoch: 5         Batch: 1         Training Loss: 2.786750
Epoch: 5         Batch: 101         Training Loss: 2.002395
Epoch: 5         Batch: 201         Training Loss: 2.019095
Epoch: 5         Batch: 301         Training Loss: 1.968766
Epoch: 5         Batch: 401         Training Loss: 1.972471
Epoch: 5         Batch: 501         Training Loss: 1.960163
Epoch: 5         Batch: 601         Training Loss: 1.957953
Epoch: 5         Batch: 701         Training Loss: 1.960859
Epoch: 5         Training Loss: 1.9660         Validation Loss: 0.9232
BOOM! Validation loss decreased (1.0137 --> 0.9232).  Saving model...
"""
```

[ ]:
```
import pickle
from os.path import exists
pickleFilname = f'/content/drive/MyDrive/College Junior/CS 4375/
 ↪model_transferAllImages{n_epochs}epochs.pkl'

if( exists(pickleFilname ) ):
    with open(pickleFilname , 'rb') as f:
        model_transfer= pickle.load(f)

with open(pickleFilname , 'wb') as f:
    pickle.dump(model_transfer, f)
```

[ ]:
```
# model_transfer
from PIL import Image

if (False): #this is a testing cell
  #filePath = '/content/dog-breed-identification/train/
 ↪0a3f1898556115d6d0931294876cd1d9.jpg' = maltese dog   3 epoch, 1000 images
  #filePath = '/content/dog-breed-identification/train/
 ↪0b97116ed04c8f0f7eb4a2b4b2620476.jpg' = samoyed (ish) 3 epoch, 1000 images
  filePath = '/content/dog-breed-identification/train/
 ↪0d103ca7cf575757374f8f6ae87d8868.jpg' #= miniature_pinscher 3 epoch, 10222␣
 ↪images
  img_dog = Image.open(filePath).convert('RGB')
  from torchvision import transforms

  # Create a preprocessing pipeline
  preprocess = transforms.Compose([
          transforms.Resize(256),
          transforms.CenterCrop(224),
          transforms.ToTensor(),
```

```
        transforms.Normalize(
        mean=[0.485, 0.456, 0.406],
        std=[0.229, 0.224, 0.225]
    )])

 # Pass the image for preprocessing and the image preprocessed
 # Reshape, crop, and normalize the input tensor for feeding into network for evaluation
 img_dog_preprocessed = preprocess(img_dog)
 batch_img_dog_tensor = torch.unsqueeze(img_dog_preprocessed, 0)

 prediction = model_transfer.forward(batch_img_dog_tensor)
 print(prediction)

 list(prediction)[0]
 list(list(prediction)[0])
 prediction.shape
```

[ ]:
```
if(False): #this is a testing cell
 model_transfer.eval()
 out = model_transfer(batch_img_dog_tensor)
 _, index = torch.max(out, 1)

 print("index:", index)
 percentage = torch.nn.functional.softmax(out, dim=1)[0] * 100
 print()
 print(percentage)
 newLabels = list(labels['target'])
 # Print the name along with score of the object identified by the model

 print(newLabels[index[0]], percentage[index[0]].item())

 # Print the top 5 scores along with the image label. Sort function is invoked on the torch to sort the scores.

 _, indices = torch.sort(out, descending=True)
 [(newLabels[idx], percentage[idx].item()) for idx in indices[0][:5]]
 print("predicted label:", labelnames.columns[index+1])
```

[ ]:
```
df = pd.DataFrame()
numCorrects = 0
numWrongs = 0

numImages = 100
for id in listOfTest[0:numImages]:
 filePath = f"/content/dog-breed-identification/train/{id}.jpg"
 df, bSuccess = identifyDog(filePath, df)
```

```
  if(bSuccess):
    numCorrects += 1
  else:
    numWrongs += 1

accuracy = numCorrects/(numCorrects+numWrongs)
print("Number of correct predictions: ", numCorrects)
print("Number of incorrect predictions: ", numWrongs)
print(f"Accuracy of the model for {numImages} dogs: ","{:.2f}%".
 ↪format(accuracy*100))
print(df.to_string())
```

All three models learn extremely slowly, though they do not seem to have hit a plateau, and the major constraints for us were time and resources. The sequential network hardly seems salvageable, as it is far too simple to capture the complexities of the cute dog pictures. The convolutional networks seemed much more promising at first, but due to the insane runtimes they were difficult to tune further. An increase in dropout seemed to marginally help the sequential network. The first run of the convolutional network had terribly slow but steady progress, so the learning rate was amplified; this changed hardly anything. In both cases, the model was only able to learn a handful of the target classes and did not reach into the full pool of one-hundred twenty breeds when making predictions, a sign of significant bias and low variance.

[1]:
```
!apt update
!apt install texlive-xetex texlive-fonts-recommended texlive-generic-recommended

import re, pathlib, shutil

# Get a list of all your Notebooks
notebooks = [x for x in pathlib.Path("/content/drive/My Drive/Colab Notebooks").
 ↪iterdir() if
             re.search(r"\.ipynb", x.name, flags = re.I)]

for i, n in enumerate(notebooks):
    print(f"\nProcessing  [{i+1:{len(str(len(notebooks)))}d}/{len(notebooks)}] ␣
 ↪{n.name}\n")

    # Optionally copy your notebooks from gdrive to your vm
    shutil.copy(n, n.name)
    n = pathlib.Path(n.name)

    !jupyter nbconvert "{n.as_posix()}" --to pdf --output "{n.stem.replace(" ",␣
 ↪"_")}"
```

```
Get:1 https://cloud.r-project.org/bin/linux/ubuntu bionic-cran40/ InRelease
[3,626 B]
Get:2 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Ign:3 https://developer.download.nvidia.com/compute/machine-
```

```
learning/repos/ubuntu1804/x86_64  InRelease
Hit:4 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64
InRelease
Hit:5 https://developer.download.nvidia.com/compute/machine-
learning/repos/ubuntu1804/x86_64  Release
Hit:6 http://archive.ubuntu.com/ubuntu bionic InRelease
Get:7 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic InRelease
[15.9 kB]
Get:8 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Hit:9 http://ppa.launchpad.net/cran/libgit2/ubuntu bionic InRelease
Get:10 http://archive.ubuntu.com/ubuntu bionic-backports InRelease [83.3 kB]
Hit:11 http://ppa.launchpad.net/deadsnakes/ppa/ubuntu bionic InRelease
Get:12 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic InRelease
[21.3 kB]
Get:14 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic/main Sources
[2,231 kB]
Get:15 http://security.ubuntu.com/ubuntu bionic-security/restricted amd64
Packages [1,307 kB]
Get:16 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic/main amd64
Packages [1,142 kB]
Get:17 http://security.ubuntu.com/ubuntu bionic-security/multiverse amd64
Packages [22.9 kB]
Get:18 http://archive.ubuntu.com/ubuntu bionic-updates/restricted amd64 Packages
[1,347 kB]
Get:19 http://archive.ubuntu.com/ubuntu bionic-updates/multiverse amd64 Packages
[30.0 kB]
Get:20 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages
[3,519 kB]
Get:21 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic/main amd64
Packages [40.8 kB]
Fetched 9,941 kB in 10s (1,024 kB/s)
Reading package lists… Done
Building dependency tree
Reading state information… Done
8 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists… Done
Building dependency tree
Reading state information… Done
The following package was automatically installed and is no longer required:
  libnvidia-common-460
Use 'apt autoremove' to remove it.
The following additional packages will be installed:
  fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono fonts-texgyre
  javascript-common libcupsfilters1 libcupsimage2 libgs9 libgs9-common
  libijs-0.35 libjbig2dec0 libjs-jquery libkpathsea6 libpotrace0 libptexenc1
  libruby2.5 libsynctex1 libtexlua52 libtexluajit2 libzzip-0-13 lmodern
  poppler-data preview-latex-style rake ruby ruby-did-you-mean ruby-minitest
  ruby-net-telnet ruby-power-assert ruby-test-unit ruby2.5
```

```
  rubygems-integration t1utils tex-common tex-gyre texlive-base
  texlive-binaries texlive-latex-base texlive-latex-extra
  texlive-latex-recommended texlive-pictures texlive-plain-generic tipa
Suggested packages:
  fonts-noto apache2 | lighttpd | httpd poppler-utils ghostscript
  fonts-japanese-mincho | fonts-ipafont-mincho fonts-japanese-gothic
  | fonts-ipafont-gothic fonts-arphic-ukai fonts-arphic-uming fonts-nanum ri
  ruby-dev bundler debhelper gv | postscript-viewer perl-tk xpdf-reader
  | pdf-viewer texlive-fonts-recommended-doc texlive-latex-base-doc
  python-pygments icc-profiles libfile-which-perl
  libspreadsheet-parseexcel-perl texlive-latex-extra-doc
  texlive-latex-recommended-doc texlive-pstricks dot2tex prerex ruby-tcltk
  | libtcltk-ruby texlive-pictures-doc vprerex
The following NEW packages will be installed:
  fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono fonts-texgyre
  javascript-common libcupsfilters1 libcupsimage2 libgs9 libgs9-common
  libijs-0.35 libjbig2dec0 libjs-jquery libkpathsea6 libpotrace0 libptexenc1
  libruby2.5 libsynctex1 libtexlua52 libtexluajit2 libzzip-0-13 lmodern
  poppler-data preview-latex-style rake ruby ruby-did-you-mean ruby-minitest
  ruby-net-telnet ruby-power-assert ruby-test-unit ruby2.5
  rubygems-integration t1utils tex-common tex-gyre texlive-base
  texlive-binaries texlive-fonts-recommended texlive-generic-recommended
  texlive-latex-base texlive-latex-extra texlive-latex-recommended
  texlive-pictures texlive-plain-generic texlive-xetex tipa
0 upgraded, 47 newly installed, 0 to remove and 8 not upgraded.
Need to get 146 MB of archives.
After this operation, 460 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic/main amd64 fonts-droid-fallback
all 1:6.0.1r16-1.1 [1,805 kB]
Get:2 http://archive.ubuntu.com/ubuntu bionic/main amd64 fonts-lato all 2.0-2
[2,698 kB]
Get:3 http://archive.ubuntu.com/ubuntu bionic/main amd64 poppler-data all
0.4.8-2 [1,479 kB]
Get:4 http://archive.ubuntu.com/ubuntu bionic/main amd64 tex-common all 6.09
[33.0 kB]
Get:5 http://archive.ubuntu.com/ubuntu bionic/main amd64 fonts-lmodern all
2.004.5-3 [4,551 kB]
Get:6 http://archive.ubuntu.com/ubuntu bionic/main amd64 fonts-noto-mono all
20171026-2 [75.5 kB]
Get:7 http://archive.ubuntu.com/ubuntu bionic/universe amd64 fonts-texgyre all
20160520-1 [8,761 kB]
Get:8 http://archive.ubuntu.com/ubuntu bionic/main amd64 javascript-common all
11 [6,066 B]
Get:9 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libcupsfilters1
amd64 1.20.2-0ubuntu3.1 [108 kB]
Get:10 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libcupsimage2
amd64 2.2.7-1ubuntu2.9 [18.6 kB]
Get:11 http://archive.ubuntu.com/ubuntu bionic/main amd64 libijs-0.35 amd64
```

```
0.35-13 [15.5 kB]
Get:12 http://archive.ubuntu.com/ubuntu bionic/main amd64 libjbig2dec0 amd64
0.13-6 [55.9 kB]
Get:13 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libgs9-common
all 9.26~dfsg+0-0ubuntu0.18.04.17 [5,092 kB]
Get:14 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libgs9 amd64
9.26~dfsg+0-0ubuntu0.18.04.17 [2,267 kB]
Get:15 http://archive.ubuntu.com/ubuntu bionic/main amd64 libjs-jquery all
3.2.1-1 [152 kB]
Get:16 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libkpathsea6
amd64 2017.20170613.44572-8ubuntu0.1 [54.9 kB]
Get:17 http://archive.ubuntu.com/ubuntu bionic/main amd64 libpotrace0 amd64
1.14-2 [17.4 kB]
Get:18 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libptexenc1
amd64 2017.20170613.44572-8ubuntu0.1 [34.5 kB]
Get:19 http://archive.ubuntu.com/ubuntu bionic/main amd64 rubygems-integration
all 1.11 [4,994 B]
Get:20 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 ruby2.5 amd64
2.5.1-1ubuntu1.12 [48.6 kB]
Get:21 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby amd64 1:2.5.1
[5,712 B]
Get:22 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 rake all
12.3.1-1ubuntu0.1 [44.9 kB]
Get:23 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-did-you-mean all
1.2.0-2 [9,700 B]
Get:24 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-minitest all
5.10.3-1 [38.6 kB]
Get:25 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-net-telnet all
0.1.1-2 [12.6 kB]
Get:26 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-power-assert all
0.3.0-1 [7,952 B]
Get:27 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-test-unit all
3.2.5-1 [61.1 kB]
Get:28 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libruby2.5
amd64 2.5.1-1ubuntu1.12 [3,073 kB]
Get:29 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libsynctex1
amd64 2017.20170613.44572-8ubuntu0.1 [41.4 kB]
Get:30 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libtexlua52
amd64 2017.20170613.44572-8ubuntu0.1 [91.2 kB]
Get:31 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libtexluajit2
amd64 2017.20170613.44572-8ubuntu0.1 [230 kB]
Get:32 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libzzip-0-13
amd64 0.13.62-3.1ubuntu0.18.04.1 [26.0 kB]
Get:33 http://archive.ubuntu.com/ubuntu bionic/main amd64 lmodern all 2.004.5-3
[9,631 kB]
Get:34 http://archive.ubuntu.com/ubuntu bionic/main amd64 preview-latex-style
all 11.91-1ubuntu1 [185 kB]
Get:35 http://archive.ubuntu.com/ubuntu bionic/main amd64 t1utils amd64 1.41-2
```

```
[56.0 kB]
Get:36 http://archive.ubuntu.com/ubuntu bionic/universe amd64 tex-gyre all
20160520-1 [4,998 kB]
Get:37 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 texlive-
binaries amd64 2017.20170613.44572-8ubuntu0.1 [8,179 kB]
Get:38 http://archive.ubuntu.com/ubuntu bionic/main amd64 texlive-base all
2017.20180305-1 [18.7 MB]
Get:39 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-fonts-
recommended all 2017.20180305-1 [5,262 kB]
Get:40 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-plain-
generic all 2017.20180305-2 [23.6 MB]
Get:41 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-generic-
recommended all 2017.20180305-1 [15.9 kB]
Get:42 http://archive.ubuntu.com/ubuntu bionic/main amd64 texlive-latex-base all
2017.20180305-1 [951 kB]
Get:43 http://archive.ubuntu.com/ubuntu bionic/main amd64 texlive-latex-
recommended all 2017.20180305-1 [14.9 MB]
Get:44 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-pictures
all 2017.20180305-1 [4,026 kB]
Get:45 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-latex-
extra all 2017.20180305-2 [10.6 MB]
Get:46 http://archive.ubuntu.com/ubuntu bionic/universe amd64 tipa all 2:1.3-20
[2,978 kB]
Get:47 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-xetex all
2017.20180305-1 [10.7 MB]
Fetched 146 MB in 8s (18.3 MB/s)
Extracting templates from packages: 100%
Preconfiguring packages …
Selecting previously unselected package fonts-droid-fallback.
(Reading database … 124015 files and directories currently installed.)
Preparing to unpack …/00-fonts-droid-fallback_1%3a6.0.1r16-1.1_all.deb …
Unpacking fonts-droid-fallback (1:6.0.1r16-1.1) …
Selecting previously unselected package fonts-lato.
Preparing to unpack …/01-fonts-lato_2.0-2_all.deb …
Unpacking fonts-lato (2.0-2) …
Selecting previously unselected package poppler-data.
Preparing to unpack …/02-poppler-data_0.4.8-2_all.deb …
Unpacking poppler-data (0.4.8-2) …
Selecting previously unselected package tex-common.
Preparing to unpack …/03-tex-common_6.09_all.deb …
Unpacking tex-common (6.09) …
Selecting previously unselected package fonts-lmodern.
Preparing to unpack …/04-fonts-lmodern_2.004.5-3_all.deb …
Unpacking fonts-lmodern (2.004.5-3) …
Selecting previously unselected package fonts-noto-mono.
Preparing to unpack …/05-fonts-noto-mono_20171026-2_all.deb …
Unpacking fonts-noto-mono (20171026-2) …
Selecting previously unselected package fonts-texgyre.
```

```
Preparing to unpack …/06-fonts-texgyre_20160520-1_all.deb …
Unpacking fonts-texgyre (20160520-1) …
Selecting previously unselected package javascript-common.
Preparing to unpack …/07-javascript-common_11_all.deb …
Unpacking javascript-common (11) …
Selecting previously unselected package libcupsfilters1:amd64.
Preparing to unpack …/08-libcupsfilters1_1.20.2-0ubuntu3.1_amd64.deb …
Unpacking libcupsfilters1:amd64 (1.20.2-0ubuntu3.1) …
Selecting previously unselected package libcupsimage2:amd64.
Preparing to unpack …/09-libcupsimage2_2.2.7-1ubuntu2.9_amd64.deb …
Unpacking libcupsimage2:amd64 (2.2.7-1ubuntu2.9) …
Selecting previously unselected package libijs-0.35:amd64.
Preparing to unpack …/10-libijs-0.35_0.35-13_amd64.deb …
Unpacking libijs-0.35:amd64 (0.35-13) …
Selecting previously unselected package libjbig2dec0:amd64.
Preparing to unpack …/11-libjbig2dec0_0.13-6_amd64.deb …
Unpacking libjbig2dec0:amd64 (0.13-6) …
Selecting previously unselected package libgs9-common.
Preparing to unpack …/12-libgs9-common_9.26~dfsg+0-0ubuntu0.18.04.17_all.deb
…
Unpacking libgs9-common (9.26~dfsg+0-0ubuntu0.18.04.17) …
Selecting previously unselected package libgs9:amd64.
Preparing to unpack …/13-libgs9_9.26~dfsg+0-0ubuntu0.18.04.17_amd64.deb …
Unpacking libgs9:amd64 (9.26~dfsg+0-0ubuntu0.18.04.17) …
Selecting previously unselected package libjs-jquery.
Preparing to unpack …/14-libjs-jquery_3.2.1-1_all.deb …
Unpacking libjs-jquery (3.2.1-1) …
Selecting previously unselected package libkpathsea6:amd64.
Preparing to unpack …/15-libkpathsea6_2017.20170613.44572-8ubuntu0.1_amd64.deb
…
Unpacking libkpathsea6:amd64 (2017.20170613.44572-8ubuntu0.1) …
Selecting previously unselected package libpotrace0.
Preparing to unpack …/16-libpotrace0_1.14-2_amd64.deb …
Unpacking libpotrace0 (1.14-2) …
Selecting previously unselected package libptexenc1:amd64.
Preparing to unpack …/17-libptexenc1_2017.20170613.44572-8ubuntu0.1_amd64.deb
…
Unpacking libptexenc1:amd64 (2017.20170613.44572-8ubuntu0.1) …
Selecting previously unselected package rubygems-integration.
Preparing to unpack …/18-rubygems-integration_1.11_all.deb …
Unpacking rubygems-integration (1.11) …
Selecting previously unselected package ruby2.5.
Preparing to unpack …/19-ruby2.5_2.5.1-1ubuntu1.12_amd64.deb …
Unpacking ruby2.5 (2.5.1-1ubuntu1.12) …
Selecting previously unselected package ruby.
Preparing to unpack …/20-ruby_1%3a2.5.1_amd64.deb …
Unpacking ruby (1:2.5.1) …
Selecting previously unselected package rake.
```

```
Preparing to unpack …/21-rake_12.3.1-1ubuntu0.1_all.deb …
Unpacking rake (12.3.1-1ubuntu0.1) …
Selecting previously unselected package ruby-did-you-mean.
Preparing to unpack …/22-ruby-did-you-mean_1.2.0-2_all.deb …
Unpacking ruby-did-you-mean (1.2.0-2) …
Selecting previously unselected package ruby-minitest.
Preparing to unpack …/23-ruby-minitest_5.10.3-1_all.deb …
Unpacking ruby-minitest (5.10.3-1) …
Selecting previously unselected package ruby-net-telnet.
Preparing to unpack …/24-ruby-net-telnet_0.1.1-2_all.deb …
Unpacking ruby-net-telnet (0.1.1-2) …
Selecting previously unselected package ruby-power-assert.
Preparing to unpack …/25-ruby-power-assert_0.3.0-1_all.deb …
Unpacking ruby-power-assert (0.3.0-1) …
Selecting previously unselected package ruby-test-unit.
Preparing to unpack …/26-ruby-test-unit_3.2.5-1_all.deb …
Unpacking ruby-test-unit (3.2.5-1) …
Selecting previously unselected package libruby2.5:amd64.
Preparing to unpack …/27-libruby2.5_2.5.1-1ubuntu1.12_amd64.deb …
Unpacking libruby2.5:amd64 (2.5.1-1ubuntu1.12) …
Selecting previously unselected package libsynctex1:amd64.
Preparing to unpack …/28-libsynctex1_2017.20170613.44572-8ubuntu0.1_amd64.deb
…
Unpacking libsynctex1:amd64 (2017.20170613.44572-8ubuntu0.1) …
Selecting previously unselected package libtexlua52:amd64.
Preparing to unpack …/29-libtexlua52_2017.20170613.44572-8ubuntu0.1_amd64.deb
…
Unpacking libtexlua52:amd64 (2017.20170613.44572-8ubuntu0.1) …
Selecting previously unselected package libtexluajit2:amd64.
Preparing to unpack
…/30-libtexluajit2_2017.20170613.44572-8ubuntu0.1_amd64.deb …
Unpacking libtexluajit2:amd64 (2017.20170613.44572-8ubuntu0.1) …
Selecting previously unselected package libzzip-0-13:amd64.
Preparing to unpack …/31-libzzip-0-13_0.13.62-3.1ubuntu0.18.04.1_amd64.deb …
Unpacking libzzip-0-13:amd64 (0.13.62-3.1ubuntu0.18.04.1) …
Selecting previously unselected package lmodern.
Preparing to unpack …/32-lmodern_2.004.5-3_all.deb …
Unpacking lmodern (2.004.5-3) …
Selecting previously unselected package preview-latex-style.
Preparing to unpack …/33-preview-latex-style_11.91-1ubuntu1_all.deb …
Unpacking preview-latex-style (11.91-1ubuntu1) …
Selecting previously unselected package t1utils.
Preparing to unpack …/34-t1utils_1.41-2_amd64.deb …
Unpacking t1utils (1.41-2) …
Selecting previously unselected package tex-gyre.
Preparing to unpack …/35-tex-gyre_20160520-1_all.deb …
Unpacking tex-gyre (20160520-1) …
Selecting previously unselected package texlive-binaries.
```

```
Preparing to unpack …/36-texlive-
binaries_2017.20170613.44572-8ubuntu0.1_amd64.deb …
Unpacking texlive-binaries (2017.20170613.44572-8ubuntu0.1) …
Selecting previously unselected package texlive-base.
Preparing to unpack …/37-texlive-base_2017.20180305-1_all.deb …
Unpacking texlive-base (2017.20180305-1) …
Selecting previously unselected package texlive-fonts-recommended.
Preparing to unpack …/38-texlive-fonts-recommended_2017.20180305-1_all.deb …
Unpacking texlive-fonts-recommended (2017.20180305-1) …
Selecting previously unselected package texlive-plain-generic.
Preparing to unpack …/39-texlive-plain-generic_2017.20180305-2_all.deb …
Unpacking texlive-plain-generic (2017.20180305-2) …
Selecting previously unselected package texlive-generic-recommended.
Preparing to unpack …/40-texlive-generic-recommended_2017.20180305-1_all.deb
…
Unpacking texlive-generic-recommended (2017.20180305-1) …
Selecting previously unselected package texlive-latex-base.
Preparing to unpack …/41-texlive-latex-base_2017.20180305-1_all.deb …
Unpacking texlive-latex-base (2017.20180305-1) …
Selecting previously unselected package texlive-latex-recommended.
Preparing to unpack …/42-texlive-latex-recommended_2017.20180305-1_all.deb …
Unpacking texlive-latex-recommended (2017.20180305-1) …
Selecting previously unselected package texlive-pictures.
Preparing to unpack …/43-texlive-pictures_2017.20180305-1_all.deb …
Unpacking texlive-pictures (2017.20180305-1) …
Selecting previously unselected package texlive-latex-extra.
Preparing to unpack …/44-texlive-latex-extra_2017.20180305-2_all.deb …
Unpacking texlive-latex-extra (2017.20180305-2) …
Selecting previously unselected package tipa.
Preparing to unpack …/45-tipa_2%3a1.3-20_all.deb …
Unpacking tipa (2:1.3-20) …
Selecting previously unselected package texlive-xetex.
Preparing to unpack …/46-texlive-xetex_2017.20180305-1_all.deb …
Unpacking texlive-xetex (2017.20180305-1) …
Setting up libgs9-common (9.26~dfsg+0-0ubuntu0.18.04.17) …
Setting up libkpathsea6:amd64 (2017.20170613.44572-8ubuntu0.1) …
Setting up libjs-jquery (3.2.1-1) …
Setting up libtexlua52:amd64 (2017.20170613.44572-8ubuntu0.1) …
Setting up fonts-droid-fallback (1:6.0.1r16-1.1) …
Setting up libsynctex1:amd64 (2017.20170613.44572-8ubuntu0.1) …
Setting up libptexenc1:amd64 (2017.20170613.44572-8ubuntu0.1) …
Setting up tex-common (6.09) …
update-language: texlive-base not installed and configured, doing nothing!
Setting up poppler-data (0.4.8-2) …
Setting up tex-gyre (20160520-1) …
Setting up preview-latex-style (11.91-1ubuntu1) …
Setting up fonts-texgyre (20160520-1) …
Setting up fonts-noto-mono (20171026-2) …
```

```
Setting up fonts-lato (2.0-2) …
Setting up libcupsfilters1:amd64 (1.20.2-0ubuntu3.1) …
Setting up libcupsimage2:amd64 (2.2.7-1ubuntu2.9) …
Setting up libjbig2dec0:amd64 (0.13-6) …
Setting up ruby-did-you-mean (1.2.0-2) …
Setting up t1utils (1.41-2) …
Setting up ruby-net-telnet (0.1.1-2) …
Setting up libijs-0.35:amd64 (0.35-13) …
Setting up rubygems-integration (1.11) …
Setting up libpotrace0 (1.14-2) …
Setting up javascript-common (11) …
Setting up ruby-minitest (5.10.3-1) …
Setting up libzzip-0-13:amd64 (0.13.62-3.1ubuntu0.18.04.1) …
Setting up libgs9:amd64 (9.26~dfsg+0-0ubuntu0.18.04.17) …
Setting up libtexluajit2:amd64 (2017.20170613.44572-8ubuntu0.1) …
Setting up fonts-lmodern (2.004.5-3) …
Setting up ruby-power-assert (0.3.0-1) …
Setting up texlive-binaries (2017.20170613.44572-8ubuntu0.1) …
update-alternatives: using /usr/bin/xdvi-xaw to provide /usr/bin/xdvi.bin
(xdvi.bin) in auto mode
update-alternatives: using /usr/bin/bibtex.original to provide /usr/bin/bibtex
(bibtex) in auto mode
Setting up texlive-base (2017.20180305-1) …
mktexlsr: Updating /var/lib/texmf/ls-R-TEXLIVEDIST…
mktexlsr: Updating /var/lib/texmf/ls-R-TEXMFMAIN…
mktexlsr: Updating /var/lib/texmf/ls-R…
mktexlsr: Done.
tl-paper: setting paper size for dvips to a4:
/var/lib/texmf/dvips/config/config-paper.ps
tl-paper: setting paper size for dvipdfmx to a4:
/var/lib/texmf/dvipdfmx/dvipdfmx-paper.cfg
tl-paper: setting paper size for xdvi to a4: /var/lib/texmf/xdvi/XDvi-paper
tl-paper: setting paper size for pdftex to a4:
/var/lib/texmf/tex/generic/config/pdftexconfig.tex
Setting up texlive-fonts-recommended (2017.20180305-1) …
Setting up texlive-plain-generic (2017.20180305-2) …
Setting up texlive-generic-recommended (2017.20180305-1) …
Setting up texlive-latex-base (2017.20180305-1) …
Setting up lmodern (2.004.5-3) …
Setting up texlive-latex-recommended (2017.20180305-1) …
Setting up texlive-pictures (2017.20180305-1) …
Setting up tipa (2:1.3-20) …
Regenerating '/var/lib/texmf/fmtutil.cnf-DEBIAN'… done.
Regenerating '/var/lib/texmf/fmtutil.cnf-TEXLIVEDIST'… done.
update-fmtutil has updated the following file(s):
        /var/lib/texmf/fmtutil.cnf-DEBIAN
        /var/lib/texmf/fmtutil.cnf-TEXLIVEDIST
If you want to activate the changes in the above file(s),
```

```
you should run fmtutil-sys or fmtutil.
Setting up texlive-latex-extra (2017.20180305-2) …
Setting up texlive-xetex (2017.20180305-1) …
Setting up ruby2.5 (2.5.1-1ubuntu1.12) …
Setting up ruby (1:2.5.1) …
Setting up ruby-test-unit (3.2.5-1) …
Setting up rake (12.3.1-1ubuntu0.1) …
Setting up libruby2.5:amd64 (2.5.1-1ubuntu1.12) …
Processing triggers for mime-support (3.60ubuntu1) …
Processing triggers for libc-bin (2.27-3ubuntu1.6) …
Processing triggers for man-db (2.8.3-2ubuntu0.1) …
Processing triggers for fontconfig (2.12.6-0ubuntu2) …
Processing triggers for tex-common (6.09) …
Running updmap-sys. This may take some time… done.
Running mktexlsr /var/lib/texmf … done.
Building format(s) --all.
        This may take some time… done.


      ␣
↪------------------------------------------------------------------------

       FileNotFoundError                         Traceback (most recent call␣
↪last)

       <ipython-input-1-db19b0469687> in <module>
         5
         6 # Get a list of all your Notebooks
   ----> 7 notebooks = [x for x in pathlib.Path("/content/drive/My Drive/Colab␣
↪Notebooks").iterdir() if
         8              re.search(r"\.ipynb", x.name, flags = re.I)]
         9


       <ipython-input-1-db19b0469687> in <listcomp>(.0)
         5
         6 # Get a list of all your Notebooks
   ----> 7 notebooks = [x for x in pathlib.Path("/content/drive/My Drive/Colab␣
↪Notebooks").iterdir() if
         8              re.search(r"\.ipynb", x.name, flags = re.I)]
         9


       /usr/lib/python3.8/pathlib.py in iterdir(self)
      1120         if self._closed:
      1121             self._raise_closed()
   -> 1122         for name in self._accessor.listdir(self):
      1123             if name in {'.', '..'}:
```

```
      1124                 # Yielding a path object for these makes little sense


       FileNotFoundError: [Errno 2] No such file or directory: '/content/drive/
↪My Drive/Colab Notebooks'
```

```
import seaborn as sb
import matplotlib.pyplot as plt
import warnings
import pandas as pd
import numpy as np
import requests
import io

import torch
import torch.nn as nn
import torch.nn.functional as F
from torch.utils.data import DataLoader
from torchvision import datasets, transforms, models
from torchvision.utils import make_grid

from PIL import Image
from IPython.display import display
import cv2
from PIL import ImageFile
import torchvision.transforms as transforms
ImageFile.LOAD_TRUNCATED_IMAGES = True

import glob
import os
import random
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')

if(True):
  #link to google drive
  from google.colab import drive
  drive.mount('/content/drive')

if(True):
  #unzip data
  import zipfile
  path_to_zip_file = r'/content/drive/MyDrive/UTD/.UTD 2022 Fall/CS 4372/dog-breed-identification.zip'
  directory_to_extract_to = "dog-breed-identification"
  !mkdir "dog-breed-identification"
  with zipfile.ZipFile(path_to_zip_file, 'r') as zip_ref:
      zip_ref.extractall(directory_to_extract_to)


%cd "/content/dog-breed-identification"

labels = pd.read_csv('labels.csv')
labelnames = pd.read_csv('sample_submission.csv')

print(f"There are {len(labelnames)} label names")
print(labelnames.head(5).to_string())
print()
print(f"There are {len(labels)} labels")
print(labels.head(5).to_string())
```

```
# from the looks of the csv file;, it is
# given a submission [identified by id], gives probabilities [in the columns] of being [column title]
```

```
/content/dog-breed-identification
There are 10357 label names
                                  id  affenpinscher  afghan_hound  african_hunting_dog  airedale  a
0  000621fb3cbb32d8935728e48679680e       0.008333      0.008333             0.008333  0.008333
1  00102ee9d8eb90812350685311fe5890       0.008333      0.008333             0.008333  0.008333
2  0012a730dfa437f5f3613fb75efcd4ce       0.008333      0.008333             0.008333  0.008333
3  001510bc8570bbeee98c8d80c8a95ec1       0.008333      0.008333             0.008333  0.008333
4  001a5f3114548acdefa3d4da05474c2e       0.008333      0.008333             0.008333  0.008333

There are 10222 labels
                                  id             breed
0  000bec180eb18c7604dcecc8fe0dba07       boston_bull
1  001513dfcb2ffafc82cccf4d8bbaba97             dingo
2  001cdf01b096e06d78e9e5112d419397          pekinese
3  00214f311d5d2247d5dfe4fe24b2303d          bluetick
4  0021f9ceb3235effd7fcde7f7538ed62  golden_retriever
```

```
#print test images
#this cell is a testing cell
if(False):
  %cd "train"
  fileDir = "." #'dog-breed-identification/train'
  listOfCuteDogImages = [os.path.join(fileDir, _) for _ in os.listdir(fileDir)]
  #print(listOfCuteDogImages[0:3]) #print first 3 image filename

  for imageName in listOfCuteDogImages[7:11]: #plot images of 7th to 11th dog
    print (os.path.join(imageName))
    img = cv2.imread(os.path.join(imageName))
    #print(img) #this shows numbers..
    img_cvt=cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    plt.imshow(img_cvt)
    plt.show()
  %cd ..
```

```
codes = range(len(labelnames))
breed_to_code = dict(zip(labelnames, codes))
code_to_breed = dict(zip(codes, labelnames))
labels['target'] =  [breed_to_code[x] for x in labels.breed]
labels['rank'] = labels.groupby('breed').rank()['id']
#labels = labels[0:5000] #testing
labels_pivot = labels.pivot('id', 'breed', 'target').reset_index().fillna(0)

#split into training samples and testing samples (valid)
train = labels_pivot.sample(frac=0.85)
valid = labels_pivot[~labels_pivot['id'].isin(train['id'])]
print(train.shape, valid.shape)
```

```
(8689, 121) (1533, 121)
```

```
import pickle
from os.path import exists
n_epochs = 5
```

```
pickleFilname = f'/content/drive/MyDrive/UTD/.UTD 2022 Fall/CS 4372/model_transfer_{train.shape[0]}imag

if( exists(pickleFilname ) ):
    with open(pickleFilname , 'rb') as f:
        model_transfer= pickle.load(f)
```

Data

```
print(train.head(5).to_string())
listOfTrain = list(train['id']) #list training data, without ".jpg"
```

	breed	id	affenpinscher	afghan_hound	african_hunting_dog	aireda
1961	3059a265274759f6a68a553bb3ed865f		0.0	0.0	0.0	0
7187	b4173f95948cd008831eeea46ace8498		0.0	0.0	0.0	0
8826	dd28cda8dfc9a97642d3333722a520af		0.0	0.0	0.0	0
611	0ef80d6c4d739faaae8d3cc08c696c54		0.0	0.0	0.0	0
5282	852b9c2a1af49cea2fed90f8eafbf0de		0.0	0.0	0.0	0

```
print(valid.head(5).to_string())
listOfTest = list(valid['id']) #list of testing data, without ".jpg"
```

	breed	id	affenpinscher	afghan_hound	african_hunting_dog	aireda
0	000bec180eb18c7604dcecc8fe0dba07		0.0	0.0	0.0	0
12	00693b8bc2470375cc744a6391d397ec		0.0	0.0	0.0	0
19	008887054b18ba3c7601792b6a453cc3		0.0	0.0	0.0	0
20	008b1271ed1addaccf93783b39deab45		0.0	0.0	0.0	0
39	010e87fdf252645a827e37470e65e842		0.0	0.0	0.0	0

```
img_transform = {
    'valid':transforms.Compose([
        transforms.Resize(size = 256, interpolation = transforms.InterpolationMode.NEAREST),
        transforms.CenterCrop(size = 224),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406],
                             [0.229, 0.224, 0.225])
    ]),
    'train':transforms.Compose([
       transforms.RandomResizedCrop(size = 256, interpolation = transforms.InterpolationMode.NEAREST),
        transforms.RandomRotation(degrees = 30),
        transforms.ColorJitter(),
        transforms.RandomHorizontalFlip(),
        transforms.CenterCrop(size=224),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406],
                             [0.229, 0.224, 0.225])
    ]),
    'test':transforms.Compose([
        transforms.Resize(size = 256, interpolation = transforms.InterpolationMode.NEAREST),
        transforms.CenterCrop(size = 224),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406],
                             [0.229, 0.224, 0.225])
```

```
    ]),
}


class DogBreedDataset(torch.utils.data.Dataset):
    'Characterizes a dataset for PyTorch'
    def __init__(self, img_dir, label, transform):
        'Initialization'
        self.img_dir = img_dir
        self.transform = transform
        self.label = label

    def __len__(self):
        'Denotes the total number of samples'
        return len(self.label)

    def __getitem__(self, index):
        if self.label is not None:
            img_name = '{}.jpg'.format(self.label.iloc[index, 0])
            fullname = self.img_dir + img_name
            image = Image.open(fullname)
            label = self.label.iloc[index, 1:].astype('float').to_numpy()
            label = np.argmax(label)
            if self.transform:
                image = self.transform(image)
            return [image, label]


batch_size = 12
num_workers = 4
train_img = DogBreedDataset('/content/dog-breed-identification/train/', train, transform = img_transfor
valid_img = DogBreedDataset('/content/dog-breed-identification/train/', valid, transform = img_transfor


dataloaders={
    'train':torch.utils.data.DataLoader(train_img, batch_size, num_workers = num_workers, shuffle=True)
    'valid':torch.utils.data.DataLoader(valid_img, batch_size, num_workers = num_workers, shuffle=False
}


train_img.transform

     Compose(
         RandomResizedCrop(size=(256, 256), scale=(0.08, 1.0), ratio=(0.75, 1.3333),
     interpolation=nearest)
         RandomRotation(degrees=[-30.0, 30.0], interpolation=nearest, expand=False, fill=0)
         ColorJitter(brightness=None, contrast=None, saturation=None, hue=None)
         RandomHorizontalFlip(p=0.5)
         CenterCrop(size=(224, 224))
         ToTensor()
         Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
     )


use_cuda = torch.cuda.is_available()


def imshow(axis, inp):
```

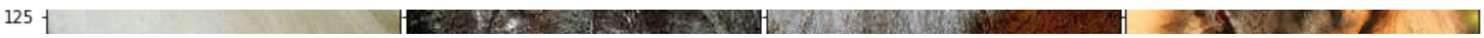
```
    """Denormalize and show"""
    inp = inp.numpy().transpose((1, 2, 0))
    mean = np.array([0.485, 0.456, 0.406])
    std = np.array([0.229, 0.224, 0.225])
    inp = std * inp + mean
    axis.imshow(inp)


type(dataloaders['train'])
```

```
     torch.utils.data.dataloader.DataLoader
```

```
from mpl_toolkits.axes_grid1 import ImageGrid
img, label = next(iter(dataloaders['train']))
print(img.size(), label.size())
fig = plt.figure(1, figsize=(16, 12))
grid = ImageGrid(fig, 111, nrows_ncols=(3, 4), axes_pad=0.05)
for i in range(img.size()[0]):
    ax = grid[i]
    imshow(ax, img[i])
```

```
torch.Size([12, 3, 224, 224]) torch.Size([12])
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] -
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] -
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] -
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] -
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] -
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] -
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] -
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] -
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] -
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] -
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] -
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] -
```

Transfer Learning and Parameter Tuning



```
##  Specify model architecture
model_transfer = models.resnet50(pretrained=True)

# Freeze training for all "features" layers
for param in model_transfer.parameters():
  #print(param)
  param.requires_grad = False

# replace the last fully connected layer with a Linnear layer 133 output
in_features = model_transfer.fc.in_features     # in_features = 2048
model_transfer.fc = nn.Linear(in_features, 120) # Linear(in_features=2048, out_features=120, bias=True)

if use_cuda:
    model_transfer = model_transfer.cuda()
```

```
criterion_transfer = nn.CrossEntropyLoss()
model_transfer_grad_paramaters = filter(lambda p: p.requires_grad, model_transfer.parameters())
optimizer_transfer = torch.optim.SGD(model_transfer_grad_paramaters, lr=0.01) #activation
```

```
# train the model
model_transfer =  train(n_epochs, dataloaders, model_transfer, optimizer_transfer, criterion_transfer,
```

```
Epoch: 1        Batch: 1        Training Loss: 4.978263
Epoch: 1        Batch: 101      Training Loss: 4.750923
Epoch: 1        Batch: 201      Training Loss: 4.603715
Epoch: 1        Batch: 301      Training Loss: 4.467474
Epoch: 1        Batch: 401      Training Loss: 4.342067
Epoch: 1        Batch: 501      Training Loss: 4.224258
Epoch: 1        Batch: 601      Training Loss: 4.117264
Epoch: 1        Batch: 701      Training Loss: 4.016170
Epoch: 1        Training Loss: 3.9962   Validation Loss: 2.7220
BOOM! Validation loss decreased (inf --> 2.7220).  Saving model...
Epoch: 2        Batch: 1        Training Loss: 3.591637
Epoch: 2        Batch: 101      Training Loss: 3.137537
Epoch: 2        Batch: 201      Training Loss: 3.082932
Epoch: 2        Batch: 301      Training Loss: 3.044538
```

```
      Epoch: 2        Batch: 401      Training Loss: 2.984761
      Epoch: 2        Batch: 501      Training Loss: 2.932749
      Epoch: 2        Batch: 601      Training Loss: 2.892359
      Epoch: 2        Batch: 701      Training Loss: 2.845057
      Epoch: 2        Training Loss: 2.8420   Validation Loss: 1.6637
      BOOM! Validation loss decreased (2.7220 --> 1.6637).  Saving model...
      Epoch: 3        Batch: 1        Training Loss: 3.061121
      Epoch: 3        Batch: 101      Training Loss: 2.515188
      Epoch: 3        Batch: 201      Training Loss: 2.461431
      Epoch: 3        Batch: 301      Training Loss: 2.429386
      Epoch: 3        Batch: 401      Training Loss: 2.410182
      Epoch: 3        Batch: 501      Training Loss: 2.388021
      Epoch: 3        Batch: 601      Training Loss: 2.370769
      Epoch: 3        Batch: 701      Training Loss: 2.355340
      Epoch: 3        Training Loss: 2.3521   Validation Loss: 1.2957
      BOOM! Validation loss decreased (1.6637 --> 1.2957).  Saving model...
      Epoch: 4        Batch: 1        Training Loss: 2.795536
      Epoch: 4        Batch: 101      Training Loss: 2.218279
      Epoch: 4        Batch: 201      Training Loss: 2.188895
      Epoch: 4        Batch: 301      Training Loss: 2.160311
      Epoch: 4        Batch: 401      Training Loss: 2.143710
      Epoch: 4        Batch: 501      Training Loss: 2.136168
      Epoch: 4        Batch: 601      Training Loss: 2.121715
      Epoch: 4        Batch: 701      Training Loss: 2.102664
      Epoch: 4        Training Loss: 2.1040   Validation Loss: 1.0825
      BOOM! Validation loss decreased (1.2957 --> 1.0825).  Saving model...
      Epoch: 5        Batch: 1        Training Loss: 2.038144
      Epoch: 5        Batch: 101      Training Loss: 1.922602
      Epoch: 5        Batch: 201      Training Loss: 1.945416
      Epoch: 5        Batch: 301      Training Loss: 1.940557
      Epoch: 5        Batch: 401      Training Loss: 1.945571
      Epoch: 5        Batch: 501      Training Loss: 1.957167
      Epoch: 5        Batch: 601      Training Loss: 1.964387
      Epoch: 5        Batch: 701      Training Loss: 1.959688
      Epoch: 5        Training Loss: 1.9605   Validation Loss: 0.8472
      BOOM! Validation loss decreased (1.0825 --> 0.8472).  Saving model...

f=""" 1000 pictures, 3 epochs:
Epoch: 1    Batch: 1    Training Loss: 4.820606
Epoch: 1    Training Loss: 4.7854   Validation Loss: 4.6281
BOOM! Validation loss decreased (inf --> 4.6281).  Saving model...
Epoch: 2    Batch: 1    Training Loss: 4.775223
Epoch: 2    Training Loss: 4.4801   Validation Loss: 4.3470
BOOM! Validation loss decreased (4.6281 --> 4.3470).  Saving model...
Epoch: 3    Batch: 1    Training Loss: 4.042047
Epoch: 3    Training Loss: 4.2297   Validation Loss: 4.0708
BOOM! Validation loss decreased (4.3470 --> 4.0708).  Saving model...
"""

f2=""" 5000 pictures, 3 epochs:
Epoch: 1    Batch: 1    Training Loss: 4.832945
Epoch: 1    Batch: 101  Training Loss: 4.741760
Epoch: 1    Batch: 201  Training Loss: 4.606566
Epoch: 1    Batch: 301  Training Loss: 4.474421
Epoch: 1    Training Loss: 4.4165   Validation Loss: 3.4888
BOOM! Validation loss decreased (inf --> 3.4888).  Saving model...
Epoch: 2    Batch: 1    Training Loss: 3.961325
Epoch: 2    Batch: 101  Training Loss: 3.838313
```

```
Epoch: 2    Batch: 201  Training Loss: 3.725471
Epoch: 2    Batch: 301  Training Loss: 3.628852
Epoch: 2    Training Loss: 3.5870   Validation Loss: 2.5023
BOOM! Validation loss decreased (3.4888 --> 2.5023).  Saving model...
Epoch: 3    Batch: 1    Training Loss: 3.083075
Epoch: 3    Batch: 101  Training Loss: 3.201915
Epoch: 3    Batch: 201  Training Loss: 3.128031
Epoch: 3    Batch: 301  Training Loss: 3.071620
Epoch: 3    Training Loss: 3.0465   Validation Loss: 1.9330
BOOM! Validation loss decreased (2.5023 --> 1.9330).  Saving model..."""

f3=""" All 10222 pictures, 3 epochs:
Epoch: 1    Batch: 1    Training Loss: 4.947224
Epoch: 1    Batch: 101  Training Loss: 4.723919
Epoch: 1    Batch: 201  Training Loss: 4.587653
Epoch: 1    Batch: 301  Training Loss: 4.468295
Epoch: 1    Batch: 401  Training Loss: 4.346153
Epoch: 1    Batch: 501  Training Loss: 4.230396
Epoch: 1    Batch: 601  Training Loss: 4.116463
Epoch: 1    Batch: 701  Training Loss: 4.006574
Epoch: 1    Training Loss: 3.9866   Validation Loss: 2.7904
BOOM! Validation loss decreased (inf --> 2.7904).  Saving model...
Epoch: 2    Batch: 1    Training Loss: 3.871949
Epoch: 2    Batch: 101  Training Loss: 3.147888
Epoch: 2    Batch: 201  Training Loss: 3.082691
Epoch: 2    Batch: 301  Training Loss: 3.044378
Epoch: 2    Batch: 401  Training Loss: 2.997859
Epoch: 2    Batch: 501  Training Loss: 2.940052
Epoch: 2    Batch: 601  Training Loss: 2.892384
Epoch: 2    Batch: 701  Training Loss: 2.848122
Epoch: 2    Training Loss: 2.8419   Validation Loss: 1.6031
BOOM! Validation loss decreased (2.7904 --> 1.6031).  Saving model...
Epoch: 3    Batch: 1    Training Loss: 2.792788
Epoch: 3    Batch: 101  Training Loss: 2.471815
Epoch: 3    Batch: 201  Training Loss: 2.474647
Epoch: 3    Batch: 301  Training Loss: 2.458736
Epoch: 3    Batch: 401  Training Loss: 2.411387
Epoch: 3    Batch: 501  Training Loss: 2.375977
Epoch: 3    Batch: 601  Training Loss: 2.349866
Epoch: 3    Batch: 701  Training Loss: 2.331016
Epoch: 3    Training Loss: 2.3327   Validation Loss: 1.2304
BOOM! Validation loss decreased (1.6031 --> 1.2304).  Saving model...
"""

f4 = """ All 10222 pictures, 5 epochs
Epoch: 1    Batch: 1    Training Loss: 4.886767
Epoch: 1    Batch: 101  Training Loss: 4.729091
Epoch: 1    Batch: 201  Training Loss: 4.608284
Epoch: 1    Batch: 301  Training Loss: 4.482468
Epoch: 1    Batch: 401  Training Loss: 4.352062
Epoch: 1    Batch: 501  Training Loss: 4.239295
Epoch: 1    Batch: 601  Training Loss: 4.125138
Epoch: 1    Batch: 701  Training Loss: 4.019838
Epoch: 1    Training Loss: 3.9988   Validation Loss: 2.6609
BOOM! Validation loss decreased (inf --> 2.6609).  Saving model...
```

```
Epoch: 2    Batch: 1    Training Loss: 3.521045
Epoch: 2    Batch: 101  Training Loss: 3.167631
Epoch: 2    Batch: 201  Training Loss: 3.127581
Epoch: 2    Batch: 301  Training Loss: 3.069848
Epoch: 2    Batch: 401  Training Loss: 3.006356
Epoch: 2    Batch: 501  Training Loss: 2.958204
Epoch: 2    Batch: 601  Training Loss: 2.906377
Epoch: 2    Batch: 701  Training Loss: 2.867462
Epoch: 2    Training Loss: 2.8605   Validation Loss: 1.6696
BOOM! Validation loss decreased (2.6609 --> 1.6696).  Saving model...
Epoch: 3    Batch: 1    Training Loss: 3.659420
Epoch: 3    Batch: 101  Training Loss: 2.491639
Epoch: 3    Batch: 201  Training Loss: 2.491406
Epoch: 3    Batch: 301  Training Loss: 2.451811
Epoch: 3    Batch: 401  Training Loss: 2.420945
Epoch: 3    Batch: 501  Training Loss: 2.407779
Epoch: 3    Batch: 601  Training Loss: 2.376457
Epoch: 3    Batch: 701  Training Loss: 2.362266
Epoch: 3    Training Loss: 2.3613   Validation Loss: 1.1551
BOOM! Validation loss decreased (1.6696 --> 1.1551).  Saving model...
Epoch: 4    Batch: 1    Training Loss: 3.070820
Epoch: 4    Batch: 101  Training Loss: 2.260064
Epoch: 4    Batch: 201  Training Loss: 2.200973
Epoch: 4    Batch: 301  Training Loss: 2.158476
Epoch: 4    Batch: 401  Training Loss: 2.162050
Epoch: 4    Batch: 501  Training Loss: 2.149412
Epoch: 4    Batch: 601  Training Loss: 2.124513
Epoch: 4    Batch: 701  Training Loss: 2.121198
Epoch: 4    Training Loss: 2.1222   Validation Loss: 1.0137
BOOM! Validation loss decreased (1.1551 --> 1.0137).  Saving model...
Epoch: 5    Batch: 1    Training Loss: 2.786750
Epoch: 5    Batch: 101  Training Loss: 2.002395
Epoch: 5    Batch: 201  Training Loss: 2.019095
Epoch: 5    Batch: 301  Training Loss: 1.968766
Epoch: 5    Batch: 401  Training Loss: 1.972471
Epoch: 5    Batch: 501  Training Loss: 1.960163
Epoch: 5    Batch: 601  Training Loss: 1.957953
Epoch: 5    Batch: 701  Training Loss: 1.960859
Epoch: 5    Training Loss: 1.9660   Validation Loss: 0.9232
BOOM! Validation loss decreased (1.0137 --> 0.9232).  Saving model...
""" #execution time: 10800s (3hours)


import pickle
from os.path import exists
pickleFilname = f'/content/drive/MyDrive/UTD/.UTD 2022 Fall/CS 4372/model_transfer_{train.shape[0]}imag

if( exists(pickleFilname ) ):
    with open(pickleFilname , 'rb') as f:
        model_transfer= pickle.load(f)

with open(pickleFilname , 'wb') as f:
    pickle.dump(model_transfer, f)


# model_transfer
```

```
from PIL import Image

if (False): #this is a testing cell
  #filePath = '/content/dog-breed-identification/train/0a3f1898556115d6d0931294876cd1d9.jpg' = maltese
  #filePath = '/content/dog-breed-identification/train/0b97116ed04c8f0f7eb4a2b4b2620476.jpg' = samoyed
  filePath = '/content/dog-breed-identification/train/0d103ca7cf575757374f8f6ae87d8868.jpg' #= miniatur
  img_dog = Image.open(filePath).convert('RGB')
  from torchvision import transforms

  # Create a preprocessing pipeline
  preprocess = transforms.Compose([
          transforms.Resize(256),
          transforms.CenterCrop(224),
          transforms.ToTensor(),
          transforms.Normalize(
          mean=[0.485, 0.456, 0.406],
          std=[0.229, 0.224, 0.225]
      )])

  # Pass the image for preprocessing and the image preprocessed
  # Reshape, crop, and normalize the input tensor for feeding into network for evaluation
  img_dog_preprocessed = preprocess(img_dog)
  batch_img_dog_tensor = torch.unsqueeze(img_dog_preprocessed, 0)

  prediction = model_transfer.forward(batch_img_dog_tensor)
  print(prediction)

  list(prediction)[0]
  list(list(prediction)[0])
  prediction.shape


if(False): #this is a testing cell
  model_transfer.eval()
  out = model_transfer(batch_img_dog_tensor)
  _, index = torch.max(out, 1)

  print("index:", index)
  percentage = torch.nn.functional.softmax(out, dim=1)[0] * 100
  print()
  print(percentage)
  newLabels = list(labels['target'])
  # Print the name along with score of the object identified by the model

  print(newLabels[index[0]], percentage[index[0]].item())

  # Print the top 5 scores along with the image label. Sort function is invoked on the torch to sort th

  _, indices = torch.sort(out, descending=True)
  [(newLabels[idx], percentage[idx].item()) for idx in indices[0][:5]]
  print("predicted label:", labelnames.columns[index+1])


def identifyDog(file, df, bShow = False):
  img_dog = Image.open(file).convert('RGB')
  #preprocessing pipeline
```

```
preprocess = transforms.Compose([
        transforms.Resize(256),
        transforms.CenterCrop(224),
        transforms.ToTensor(),
        transforms.Normalize(
        mean=[0.485, 0.456, 0.406],
        std=[0.229, 0.224, 0.225]
    )])
img_dog_preprocessed = preprocess(img_dog)
batch_img_dog_tensor = torch.unsqueeze(img_dog_preprocessed, 0)

#create prediction array
prediction = model_transfer.forward(batch_img_dog_tensor)
list(prediction)[0]
list(list(prediction)[0])

#pick out the most likely breed from predictions
model_transfer.eval()
out = model_transfer(batch_img_dog_tensor)
_, index = torch.max(out, 1)

#choose the label that matches the index
percentage = torch.nn.functional.softmax(out, dim=1)[0] * 100
newLabels = list(labels['target'])

#find actual breed in label csv
theId = file.split('/')[-1].replace('.jpg','')
theBreed = list(labels[labels['id'].str.contains(theId)]['breed'])[0]

#print prediction and actual breed
predBreed = labelnames.columns[index+1]
#print("Predicted dog breed:", predBreed)
#print("Actual dog breed:", theBreed)

dict = { "Actual Label": theBreed, "Predicted Label": predBreed, "Match": "True" if theBreed == predB
df = df.append(dict, ignore_index = True)

if(bShow):
#plot dog image
  img = cv2.imread(file)
  img_cvt=cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
  plt.imshow(img_cvt)
  plt.show()

bSuccess = True

#if prediction is wrong, plot the predicted dog
if(theBreed != predBreed):
  bSuccess = False
  if(bShow):
    #print(f"The predicted breed is wrong. Here is what a {predBreed} normally looks like.")
    #find what the predicted breed is
    thePredBreedId = list(labels[labels['breed'].str.contains(predBreed)]['id'])[0]
    #plot that dog
    file = f"/content/dog-breed-identification/train/{thePredBreedId}.jpg"
```

```
      img = cv2.imread(file)
      img_cvt=cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
      plt.imshow(img_cvt)
      plt.show()

  return df, bSuccess


df = pd.DataFrame()
numCorrects = 0
numWrongs = 0

numImages = 25
for id in listOfTest[0:numImages]:
  filePath = f"/content/dog-breed-identification/train/{id}.jpg"
  df, bSuccess = identifyDog(filePath, df)
  if(bSuccess):
    numCorrects += 1
  else:
    numWrongs += 1

accuracy = numCorrects/(numCorrects+numWrongs)
print("Number of correct predictions: ", numCorrects)
print("Number of incorrect predictions: ", numWrongs)
print(f"Accuracy of the model for {numImages} dogs: ","{:.2f}%".format(accuracy*100))
print(df.to_string())
```

```
Number of correct predictions:  18
Number of incorrect predictions:  7
Accuracy of the model for 25 dogs:  72.00%
                      Actual Label                Predicted Label  Match
0                      boston_bull                    boston_bull   True
1                      maltese_dog                    maltese_dog   True
2                            boxer                          boxer   True
3                         doberman                       doberman   True
4                      groenendael                     schipperke  False
5                     affenpinscher                            pug  False
6                   german_shepherd                german_shepherd   True
7                    english_setter                 english_setter   True
8                          papillon               blenheim_spaniel  False
9                  brittany_spaniel               brittany_spaniel   True
10              rhodesian_ridgeback                         vizsla  False
11                         leonberg                       leonberg   True
12                 mexican_hairless               mexican_hairless   True
13                    irish_terrier                            pug  False
14              rhodesian_ridgeback            rhodesian_ridgeback   True
15                         leonberg                       leonberg   True
16                         airedale                       airedale   True
17  american_staffordshire_terrier  american_staffordshire_terrier   True
18                    border_collie                  border_collie   True
19                     walker_hound                   walker_hound   True
20                   border_terrier                 norfolk_terrier  False
21               miniature_pinscher             miniature_pinscher   True
22               standard_schnauzer            miniature_schnauzer  False
23                        otterhound                     otterhound   True
24                           samoyed                        samoyed   True
```

""" 1000 train image size, with 1 epoch.
Number of correct predictions: 0
Number of incorrect predictions: 25
Accuracy of the model for 25 dogs: 0.00%

	Actual Label	Predicted Label	Match
0	dingo	bloodhound	False
1	bluetick	blenheim_spaniel	False
2	scottish_deerhound	welsh_springer_spaniel	False
3	boxer	bloodhound	False
4	boston_bull	bloodhound	False
5	affenpinscher	irish_wolfhound	False
6	border_terrier	bloodhound	False
7	irish_water_spaniel	chesapeake_bay_retriever	False
8	greater_swiss_mountain_dog	bloodhound	False
9	australian_terrier	maltese_dog	False
10	rhodesian_ridgeback	bloodhound	False
11	samoyed	irish_setter	False
12	brittany_spaniel	welsh_springer_spaniel	False
13	irish_terrier	bloodhound	False
14	pug	bloodhound	False
15	standard_schnauzer	bloodhound	False
16	rhodesian_ridgeback	bloodhound	False
17	miniature_schnauzer	maltese_dog	False
18	golden_retriever	bloodhound	False
19	old_english_sheepdog	tibetan_terrier	False
20	leonberg	welsh_springer_spaniel	False
21	standard_schnauzer	bloodhound	False
22	scottish_deerhound	welsh_springer_spaniel	False
23	american_staffordshire_terrier	bloodhound	False
24	african_hunting_dog	chesapeake_bay_retriever	False"""

""" 10222 train image size, with 3 epoch
Number of correct predictions: 15
Number of incorrect predictions: 10
Accuracy of the model for 25 dogs: 60.00%

	Actual Label	Predicted Label	Match
0	shetland_sheepdog	collie	False
1	wire-haired_fox_terrier	wire-haired_fox_terrier	True
2	boxer	boxer	True
3	otterhound	otterhound	True
4	affenpinscher	chihuahua	False
5	giant_schnauzer	giant_schnauzer	True
6	golden_retriever	golden_retriever	True
7	lakeland_terrier	lakeland_terrier	True
8	giant_schnauzer	giant_schnauzer	True
9	appenzeller	chihuahua	False
10	border_collie	border_collie	True
11	malamute	malamute	True
12	weimaraner	weimaraner	True
13	maltese_dog	chihuahua	False
14	standard_schnauzer	miniature_schnauzer	False
15	malinois	chihuahua	False
16	pug	pug	True
17	komondor	old_english_sheepdog	False
18	airedale	chihuahua	False

```
19  wire-haired_fox_terrier                airedale  False
20                    dingo                   dingo   True
21            irish_setter            irish_setter   True
22        brittany_spaniel        brittany_spaniel   True
23        golden_retriever        golden_retriever   True
24                    dhole               chihuahua  False"""

""" 10222 train image size, with 5 epoch
Number of correct predictions:  18
Number of incorrect predictions:  7
Accuracy of the model for 25 dogs:  72.00%
                    Actual Label                Predicted Label  Match
0                     boston_bull                    boston_bull   True
1                     maltese_dog                    maltese_dog   True
2                           boxer                          boxer   True
3                        doberman                       doberman   True
4                     groenendael                     schipperke  False
5                   affenpinscher                            pug  False
6                  german_shepherd                german_shepherd   True
7                   english_setter                 english_setter   True
8                         papillon               blenheim_spaniel  False
9                 brittany_spaniel               brittany_spaniel   True
10             rhodesian_ridgeback                          vizsla  False
11                        leonberg                       leonberg   True
12                mexican_hairless               mexican_hairless   True
13                   irish_terrier                            pug  False
14             rhodesian_ridgeback            rhodesian_ridgeback   True
15                        leonberg                       leonberg   True
16                        airedale                       airedale   True
17  american_staffordshire_terrier  american_staffordshire_terrier   True
18                   border_collie                  border_collie   True
19                    walker_hound                   walker_hound   True
20                  border_terrier                 norfolk_terrier  False
21              miniature_pinscher             miniature_pinscher   True
22              standard_schnauzer             miniature_schnauzer  False
23                      otterhound                     otterhound   True
24                         samoyed                        samoyed   True"""
```