Dog Identification App

Project Overview/Domain Background

The task of assigning breed to dogs from images is considered exceptionally challenging. E*ven* human would have trouble distinguishing between a Brittany and a Welsh Springer Spaniel. In this project, I created a dog identification application capable of classifying more than 100 different kinds of dog breeds.

Problem Statement

To build an app that takes user-supplied images as input and provides an estimate of the canine's breed of dog is detected in the image. If human is detected in the image, it will provide an estimate of the dog breed that is most resembling.

Metrics

Accuracy is one of the common metrics for classifiers. The model is evaluated based on accuracy as described below.

Accuracy = (TN + TP)/(TN+TP+FN+FP) = (Number of correct assessments)/Number of all assessments)

Where

TP = True Positives

TN = True Negatives

FP = False Positives

FN = False Negatives

Data Exploration

There are 13233 human images in the human dataset and 8351 dog images in the dog dataset. Data is imbalanced as there are more human images compared to dog images. Also the number of dog images for each breed are not very similar as can be seen from the below.

The number of images in train, valid, test are: 'train': 6680, 'valid': 835, 'test': 836

```
root@942bf67450f6:/data/dog_images/train# du -a | cut -d/ -f2 | sort | uniq -c | sort -nr
```

78 005.Alaskan_malamute

75 029.Border_collie

74 015.Basset_hound

72 057.Dalmatian

70 041.Bullmastiff

70 039.Bull_terrier

70 014.Basenji

68 046.Cavalier_king_charles_spaniel

67 087.Irish_terrier

67 012.Australian_shepherd

67 011.Australian_cattle_dog

67 008.American_staffordshire_terrier

66 056.Dachshund

66 036.Briard

66 032.Boston_terrier

66 023.Bernese_mountain_dog

65 076.Golden_retriever

65 044.Cane_corso

65 034.Boxer

65 027.Bloodhound

65 021.Belgian_sheepdog

65 006.American_eskimo_dog

65 001.Affenpinscher

64 115.Papillon

64 068.Flat-coated_retriever

64 042.Cairn_terrier

64 004.Akita

63 071.German_shepherd_dog

63 051.Chow_chow

63 024.Bichon_frise

63 020.Belgian_malinois

63 017.Bearded_collie

62 082.Havanese

62 061.English_cocker_spaniel

61 060.Dogue_de_bordeaux

60 079.Great_pyrenees

60 016.Beagle

59 103.Mastiff

59 090.Italian_greyhound

59 002.Afghan_hound

58 091.Japanese_chin

58 054.Collie

58 038.Brussels_griffon

57 081.Greyhound

57 031.Borzoi

55 112.Nova_scotia_duck_tolling_retriever

55 048.Chihuahua

55 047.Chesapeake_bay_retriever

54 118.Pembroke_welsh_corgi

54 089.Irish_wolfhound

54 086.Irish_setter

54 063.English_springer_spaniel

54 062.English_setter

54 045.Cardigan_welsh_corgi

54 040.Bulldog

54 035.Boykin_spaniel

53 030.Border_terrier

53 003.Airedale_terrier

52 088.Irish_water_spaniel

52 069.French_bulldog

51 124.Poodle

51 106.Newfoundland

51 097.Lakeland_terrier

51 084.Icelandic_sheepdog

51 058.Dandie_dinmont_terrier

51 055.Curly-coated_retriever

51 050.Chinese_shar-pei

51 049.Chinese_crested

51 043.Canaan_dog

51 037.Brittany

51 018.Beauceron

51 010.Anatolian_shepherd_dog

51 007.American_foxhound

50 095.Kuvasz

50 052.Clumber_spaniel

49 129.Tibetan_mastiff

49 117.Pekingese

49 101.Maltese

49 072.German_shorthaired_pointer

49 019.Bedlington_terrier

48 070.German_pinscher

48 059.Doberman_pinscher

48 053.Cocker_spaniel

48 022.Belgian_tervuren

47 107.Norfolk_terrier

47 098.Leonberger

47 083.Ibizan_hound

47 080.Greater_swiss_mountain_dog

47 013.Australian_terrier

46 109.Norwegian_elkhound

46 033.Bouvier_des_flandres

45 130.Welsh_springer_spaniel

45 123.Pomeranian

45 111.Norwich_terrier

45 094.Komondor

45 092.Keeshond

45 075.Glen_of_imaal_terrier

44 096.Labrador_retriever

44 077.Gordon_setter

43 104.Miniature_schnauzer

43 099.Lhasa_apso

43 073.German_wirehaired_pointer

43 065.Entlebucher_mountain_dog

42 127.Silky_terrier

42 074.Giant_schnauzer

42 026.Black_russian_terrier

41 078.Great_dane

40 120.Pharaoh_hound

40 113.Old_english_sheepdog

40 064.English_toy_spaniel

38 085.Irish_red_and_white_setter

38 025.Black_and_tan_coonhound

36 114.Otterhound

36 093.Kerry_blue_terrier

36 028.Bluetick_coonhound

35 125.Portuguese_water_dog

35 100.Lowchen

35 067.Finnish_spitz

35 009.American_water_spaniel

34 110.Norwegian_lundehund

34 066.Field_spaniel

33 122.Pointer

32 119.Petit_basset_griffon_vendeen

32 105.Neapolitan_mastiff

31 133.Yorkshire_terrier

31 131.Wirehaired_pointing_griffon

31 128.Smooth_fox_terrier

31 126.Saint_bernard

31 116.Parson_russell_terrier

30 102.Manchester_terrier

29 121.Plott

27 132.Xoloitzcuintli

27 108.Norwegian_buhund

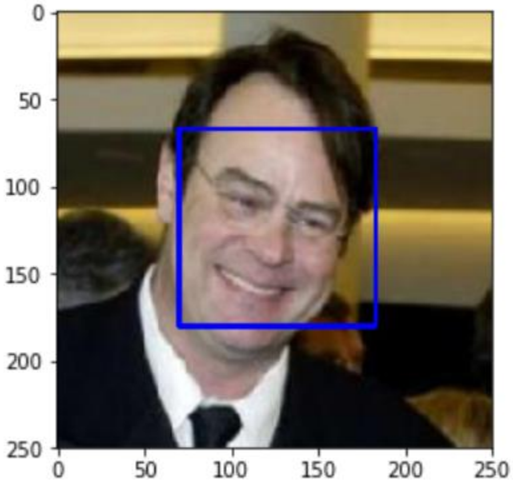
Exploratory visualization

Dog Images

Curly-Coated Retriever	American Water Spaniel

Yellow Labrador	Chocolate Labrador	Black Labrador

Human images



Algorithms and Techniques

1. CNN’s were used.
2. Initially a custom CNN architecture with 4 CNN layers and 2 FC layers was used.
    a. Drop out was applied between FC layers to avoid overfitting.
3. Then transfer learning technique was used.
    a. Pretrained resnet 50 is taken as network.

Benchmark

1. Custom CNN is trained with 4 layers and got an accuracy of 14%.

Data Preprocessing

Train data:

1. Random resized crop of 224x244 is chosen.
2. Horizontal flip is done.
3. Images are normalized with mean and standard deviation values used for pretrained models as mentioned in PyTorch documentation.

Valid & Test data:

1. Images are resized to 256x256.
2. Center crop of 224x224 is taken.
3. Images are normalized with mean and standard deviation values used for pretrained models as mentioned in PyTorch documentation.

Implementation

CNN have outperformed all classic computer vision techniques and have become one of the widely used methods to solve computer vision problems especially classification, object detection and segmentation tasks.

Created a custom CNN from scratch.

1. Has 4 conv layers with filters of 16, 32, 64, 128 respectively.

2. 2 fully connected layers with 1000, 133 nodes respectively to classify dog breeds.

3. Last fc layer has 133 nodes since we have 133 classes to distinguish from Drop out was applied between fc layers to avoid overfitting.

Refinement

Used transfer learning technique and trained a resnet50 model. Resnet is one of the classic architectures from Microsoft and have shown to do well for classification tasks.

1. Since resnet50 is trained on ImageNet dataset that has many different dog images, we can be confident that the parameters learned for resnet on ImageNet dataset would generalize well for our dataset too.

Model evaluation and validation

Custom CNN

1. Obtained test accuracy of 17% on test set after training for 20 epochs.

2. Cross entropy is used as loss function and SGD is used as optimizer.

Transfer learning resnet

1. Obtained test accuracy of 83% on test set after training for 6 epochs.

2. Cross entropy is used as loss function and Adam is used as optimizer.

3. Since resnet is of much bigger architecture, it can represent and encode complex patterns and performs well.

Justification

The transferr learned resnet obtains an accuracy of 83% which is not bad given the variation in dogs (133 classes). Also its much higher than the accuracy of 17% that we got initially.