Machine Learning Engineer Nanodegree
Dog Classifier Project

Kira Vinogradova
April 23rd, 2020

I. Definition

Project Overview

According to the database of The Fédération Cynologique Internationale is the World Canine Organisation, 368 dog breeds exist [1]. Classification of the dog breeds became important centuries ago, the breeders attempted to select dogs based on desirable characteristics and strengths. Modern dog breeds formation was driven by dog shows in the late 19th century [2].

SOME FAMOUS DOGS 1896 & 1897

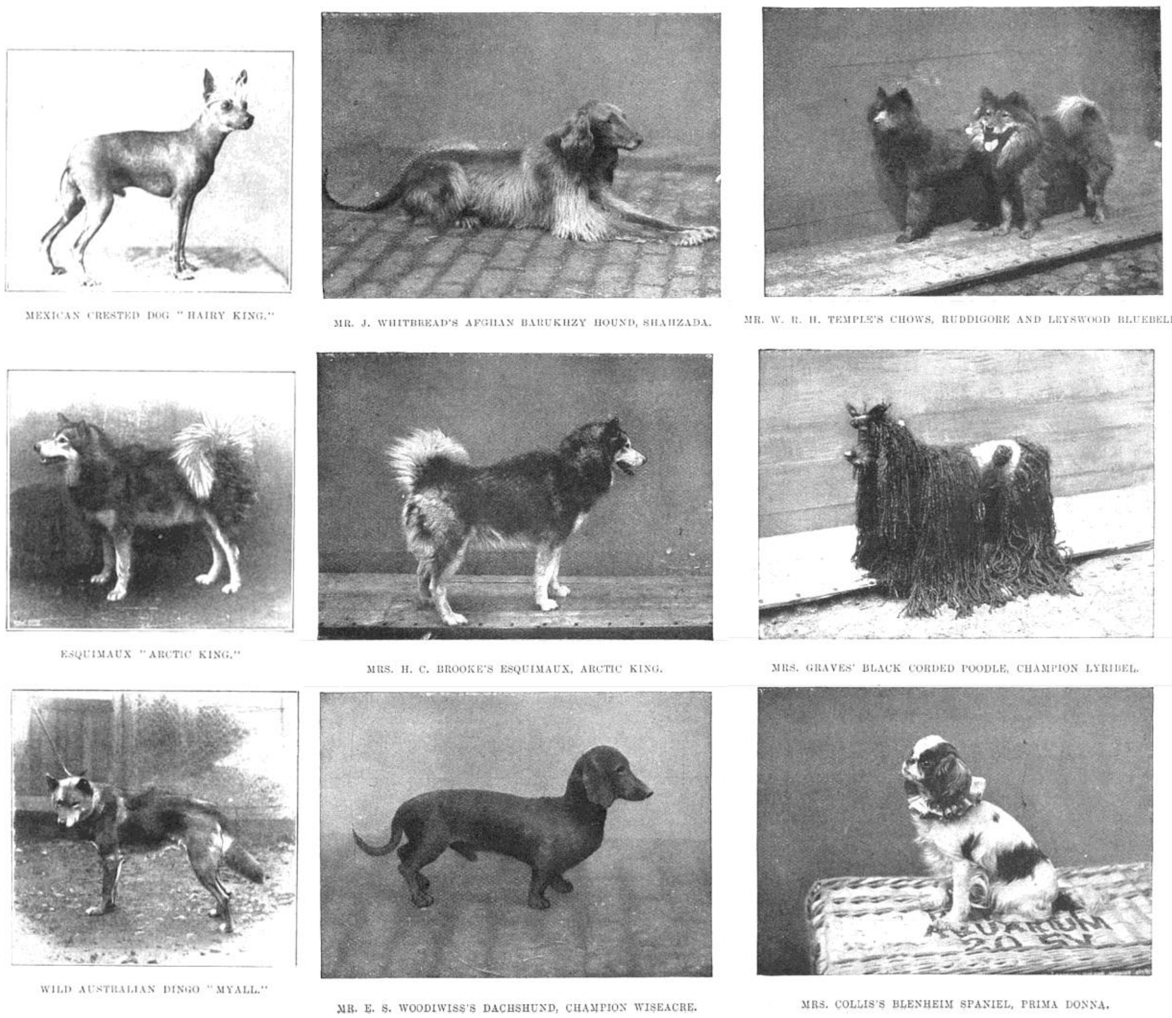


Image source: http://messybeast.com/history/1897dogs.htm

Breeds classification remains a relevant problem for the dog owners who searches for a show-class puppy or for a working dog with particular performance characteristics. Deep Learning has been proved to be suitable for image classification problems [3], therefore the task can be tackled by a Deep Learning algorithm.

Problem Statement

The primarily investigated problem is dog breed classification via Convolutional Neural Network (CNN). The project also covers the tasks:

- Dog face detection
- Human face detection

The expected behaviour of the application:

- if a dog is detected in the image, return the predicted breed.
- if a human is detected in the image, return the resembling dog breed.
- if neither is detected in the image, provide output that indicates an error.

Metrics

The performance of both models was evaluated by using the accuracy metric: number of correct predictions divided by total number of predictions.
This metric was mentioned by the exercises' designers. Accuracy is one the widely used metrics for multiclass classification evaluation.

II. Analysis

Data Exploration

Two image sets were selected for the project:

1. Dog dataset [4]. The folder contains 133 folders, each corresponding to a different dog breed.

There are 8351 total dog images.
The dataset is split into train (6680 images), test (836 im.) and validation (835 im.) sets organized in separate folders. The folders contain 133 directories, each corresponding to a different dog breed.
The images are present in various sizes and have different backgrounds. The dataset contains class unbalance: the number of items per class ranges from 4 to 8.

2. Human dataset [5] for testing on human images.

There are 13233 total human images.
The images are organized in folders by the names. There are 5750 different people (1 or more images per person). All images are the same in size - 250 by 250 pixels, but human poses and backgrounds differ.

The datasets were included into the project workspace on Udacity. Alternatively, the datasets could be downloaded via the provided links and unzipped locally.

The dog dataset was used for training, validation and testing of the CNN classifier. The human dataset was used for testing of the dog breed classifier in such case that it predicts which breed a human looks like.

Exploratory Visualization

Sample dog images:


(Alaskan_malamute_00309.jpg), (Akita_00282.jpg)

Sample human images:


(Anthony_Hopkins_0001.jpg) (Christian_Bale_0001.jpg)

Algorithms and Techniques

The solution addresses the multiclass image classification problem. Since the classified objects are images, CNNs are appropriate candidates for the algorithm. CNNs are a type of Deep Learning algorithms that can take tensors (e.g. batches of images) as an input, establish the relations between image features by learning weights and biases, output a predicted class and learn from an error.

The proposed solution, first, classifies the input images as dog, or human, or none of the two categories. Second, if a dog or a human is present on the image, the app predicts one of 133 breeds.

Humans are detected by OpenCV's implementation of pre-trained Haar feature-based cascade classifiers. Dogs are detected by trained on ImageNet VGG-16 model.

Benchmark

The solution model was compared to a benchmark CNN with:
- 4 convolutional layers, each of them followed by ReLU and 2D Max Pooling,
- flattening layer,
- drop-out layer,
- fully-connected layer,
- ReLU,
- drop-out
- fully-connected layer.

The architecture was designed to follow the common structure of CNN classifiers and VGG in particular: the first layers are convolutional layers, the number of features increases in higher layers. The feature extractor is followed by flattening of the feature tensor and the classifier: 2 dense layers with activation functions. The kernel size of (3, 3) is the most popular, having a number of features as a power of 2 is also a standard.

III. Methodology

Data Preprocessing

The input images were resized to (3,224,224) since it is the size of images from ImageNet and is expected by VGG models.
The training dataset was additionally augmented by random horizontal flips and random rotations by (-60, 60) degrees. Generally, augmentations are a powerful tool against overfitting and they help to enrich datasets. The horizontal flip was chosen because both humans and dogs are mirror-symmetric. Also, they can appear on real images in a slightly rotated pose – that is why rotations by (-60, 60) degrees were chosen.

Implementation

The implemented model consists of:

1. The base of the predictor is VGG-19 with batch normalization trained on ImageNet [6].
2. The final classification layer was replaced by another dense layer with 133 neurons instead of 1000 (the number of ImageNet classes [3]).

Refinement

The initial solution was a baseline CNN with 4 convolutional layers. It was required to be better than a random prediction with 1/133 accuracy and achieve test accuracy greater than 10%. Passing those requirements meant that the data pre-processing is reasonable.

After the data pre-processing routing was finalized, the VGG-19 with batch normalization was selected based on the comparison table of models pre-trained on ImageNet (Top-1 error) [7]. It is suitable because it has already trained feature extractor that was proved to be good at classification of ImageNet images including classification of 133 dog breeds.

The hyperparameters: the learning rate and the number of training epochs were hand-tuned during the development stage.

IV. Results

Model Evaluation and Validation

For models' comparison, the performance during training was evaluated based on the loss value on the validation set.

The implemented model was trained for 30 epochs, optimized by Stochastic Gradient Descent with the learning rate = 0.04 to reduce the Cross-entropy loss function. On the test set, the model achieved 79% of accuracy and the loss of 0.647. The achieved accuracy passes the threshold of 60%, therefore the model's performance considered to be more than sufficient, and the model is trustworthy.

The human face detector was validated on the first 100 images of human and dog datasets: 98% human faces were detected in the human subset and 17% human faces were detected in the dog subset.

The dog detector was validated on the first 100 images of human and dog datasets as well: 99% dogs were detected in the dog subset and 0% dogs were detected in the human subset.

The benchmark model was trained for 20 epochs, optimized by Stochastic Gradient Descent with the learning rate = 0.05 to reduce the Cross-entropy loss function. On the test set, the model achieved 18% of accuracy and the loss of 3.565.

Justification

The model's performance can be improved but it has already show a higher accuracy than a random classifier and the benchmark model. The final solution is capable for the dog classification problem.

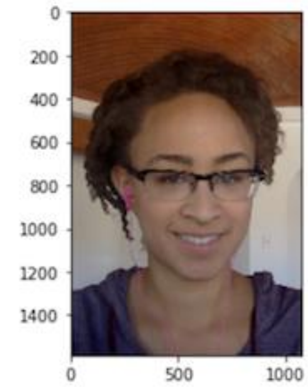
The project was tested on Udacity's workspace and on the local machine with 4.9 GB of GPU memory available for training.

V. Conclusion

Free-Form Visualization

Expected output:

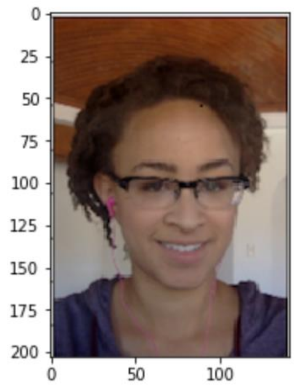
hello, human!



You look like a ...
Chinese_shar-pei

Solution:

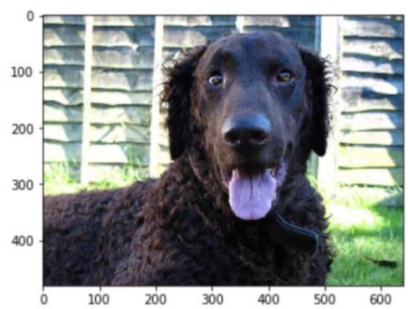
file name: ./images\sample_human_output - Copy.png



Hi, human
You look like a Kerry blue terrier

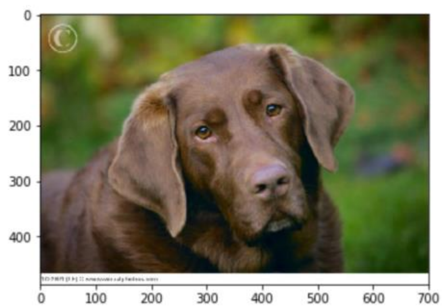
Outputs for dog images:

file name: ./images\Curly-coated_retriever_03896.jpg



Hi, dog
Your predicted breed is: Labrador retriever

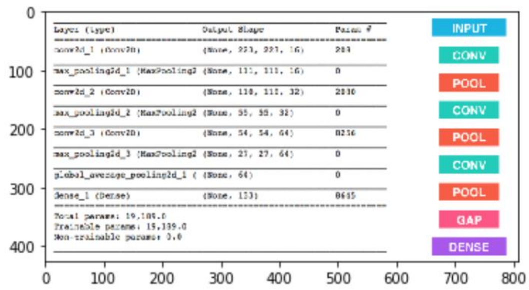
file name: ./images\Labrador_retriever_06455.jpg



Hi, dog
Your predicted breed is: Labrador retriever

Invalid input example:

file name: ./images\sample_cnn.png



No dogs or humans are detected.

Test on a mix-breed dog:

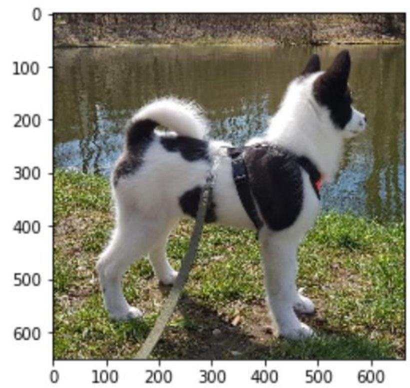
This my 11 weeks old puppy Kasy, her dad is Samoyed, her mom is Laika (her parents were Laikas but we don’t know of which particular breed) [8]. Kasy’s colour looks like a typical colour of Russian European Laika (FCI standard No. 304) but her fur texture is closer to East and West Siberian Laikas. Meanwhile, she is growing taller and heavier than pure Laika puppies.
Samoyed is one of the breeds in the training set, no types of Laika breed are present in the dog dataset.

The model predicts the Canaan dog breed for 3 of 4 images of Kasy, and the Japanese chin for 1 images. Kasy’s mom is predicted to be the Canaan dog as well. Kasy’s face and colour look like those of Canaan dogs from the Internet but her fur is of a different type and her tail is more curled.

file name: ./images\Laika.jpg

Hi, dog
Your predicted breed is: Canaan dog

file name: ./images\my_Laika_2_5mth_old.jpg



Hi, dog
Your predicted breed is: Japanese chin

Reflection

The project was conducted in 7 steps:
- Step 0: Import Dog and Human datasets
- Step 1: Detect humans by OpenCV's implementation of pre-trained Haar feature-based cascade classifiers.
- Step 2: Detect dogs by trained on ImageNet VGG-16 model.
- Step 3: Create a CNN from scratch to classify dog breeds. Train and test it.

- Step 4: Create a CNN to classify dog breeds using Transfer Learning. Train and test the solution model.
- Step 5: Design an algorithm that accepts a file path to an image and first determines whether the image contains a human, dog, or neither. Then,
  - if a dog is detected in the image, return the predicted breed.
  - if a human is detected in the image, return the resembling dog breed.
  - if neither is detected in the image, provide output that indicates an error.
- Step 6: Test the algorithm on sample images.

The hardest part of the project was step 3: find the suitable optimizer and the architecture leading to the loss' decrease.

The most enjoyable part of the project was step 6: test the app of the images of my own dog. I am curious how the predictions will change when she grows up.

Improvement

There are possible improvements:
1. Improve the model's accuracy by
   1.1. exploring more network architectures
   1.2. combining the training set with the dog images from Imagenet
   1.3. enlarging the training set by augmentations
   1.4. training the model for more epochs
   1.5. fine tuning the optimizer's parameters
2. Provide probabilities of top-5 predicted breeds
3. Explain the net's decision using interpretability techniques such as Saliency maps, Grad-CAM, Ablation-CAM, and Occlusion. It would be insightful for classification of mix-breed dogs and funny for human "breed" predictions.
4. Make a nice web app allowing to upload an image and get an instant result without running the notebook.

VI. References

[1] http://www.fci.be/en/Nomenclature/Default.aspx
[2] https://en.wikipedia.org/wiki/Dog_breed
[3] Olga Russakovsky*, Jia Deng*, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. (* = equal contribution) ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision, 2015.
[4] https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/dogImages.zip
[5] http://vis-www.cs.umass.edu/lfw/lfw.tgz
[6] Marcel Simon, Erik Rodner, Joachim Denzler, ImageNet pre-trained models with batch normalization, ArXiv, 2016.
[7] https://pytorch.org/docs/stable/torchvision/models.html#classification
[8] https://en.wikipedia.org/wiki/Laika_(dog_breed)