Machine Learning Nanodegree Capstone Report:

Dog Breed Classifier

1. Definition

Project Overview

Image classification has been an object of study since 1950s and it was a tremendous challenge because of the complexity of the task. It is vital for many current and future applications such as quality control, radiography, self-driving cars and even agriculture.

To compare different methods scientists participate in the competitions like the ImageNet Large Scale Visual Recognition Challenge or Pascal VOC before that. In 2012, a convolutional neural network (CNN) won the ImageNet Challenge for the first time and that was a breakthrough. In the following years, CNNs won every year and proved to be an excellent tool for image classification.

This breakthrough in deep neural networks became possible thanks to constant progress in CPU and especially GPU performance that we see in the last decades. Assuming this growth in computing power is highly likely to continue, CNNs will become more and more affordable and widespread, which underlines the importance of studying CNN's applications.

Another important driver of this CNN revolution is recent explosion of amount of necessary data - images that people take and share through social media. Especially pictures of their beloved dogs and cats, so this type of images are in abundance on the Web. Which gives researchers the opportunity to train their CNNs on large datasets and improve the results.

Problem Statement

The goal is to build a machine-learning model that tells us the breed of a dog on a given photo. In addition, if it is a photo of a human it tells us which dog breed it resembles the most. This model could then be used as part of an web or mobile application for real-life use when people take a photo with their phone run the image through the app and get the breed of the dog they have just met.

We have a dog breed dataset so we are doing a reinforcement learning and this dataset has 133 breeds so it is a multi-class classification problem.

This problem consists of few steps that allows us to understand different aspects of image classification and try different methods.

1. Detect humans
2. Detect dogs
3. Classify dog breeds using CNN created from scratch
4. Classify dog breeds using CNN created using transfer learning

Metrics

Accuracy is a main metric to evaluate our model.

Accuracy is the ratio of number of correct predictions to the total number of input samples.

We will also use training loss and validation loss to evaluate the training process and change model's parameters when needed.

2. Analysis

Data exploration

For this project, Udacity provided images of dogs and humans.

There are 8351 dog images divided in train, test and valid folders in 90/10/10 split. We have 133 directories (one for each dog breed) in each of train, test and valid folders. The images are of different sizes, different backgrounds, some

images are full-sized, and some are of just a head. The data is not balanced because the number of images for each breed varies.

Examples of dog images below:

The first image here has a 1024x768 resolution and the second is only 282x368. Clearly, we need to resize the images to the same resolution before we can use them to train our models.

All human photos are sorted by person's name. There are 13233 images in 5749 folders. Human images are all of the same size 250 by 250. Images are with different backgrounds and light, from different angles but just heads no full-size pictures. The data is not balanced because the number of images for each person varies.

Examples of images of humans below:

Algorithms and techniques

Convolutional Neural Networks (CNN) are the best algorithms today to classify images according to the results of the ImageNet Challenge. CNN have a significantly different architecture than other artificial neural networks. CNNs use a concept of hierarchy where first convolutional layers detects the most basic elements of an image and pass that knowledge along to the next layer. Then each next layer detects more and more complicated features of the picture.

This hierarchical nature of CNN allows us to build a model in two possible ways: from scratch or using pre-trained model. The latter is called transfer learning and it is particularly interesting because training a big CNN from scratch could be very computationally intensive. State-of-the-art CNNs, competing in the ImageNet Challenge are usually very large and require super-computers to train. So, the ability to use such pre-trained model and train only few last layers of the network to solve our specific problem is very promising.

Benchmark Model

The CNN model created from scratch must have accuracy of at least 10%.
The CNN model created using transfer learning must have accuracy of 60% and more.

3. Methodology

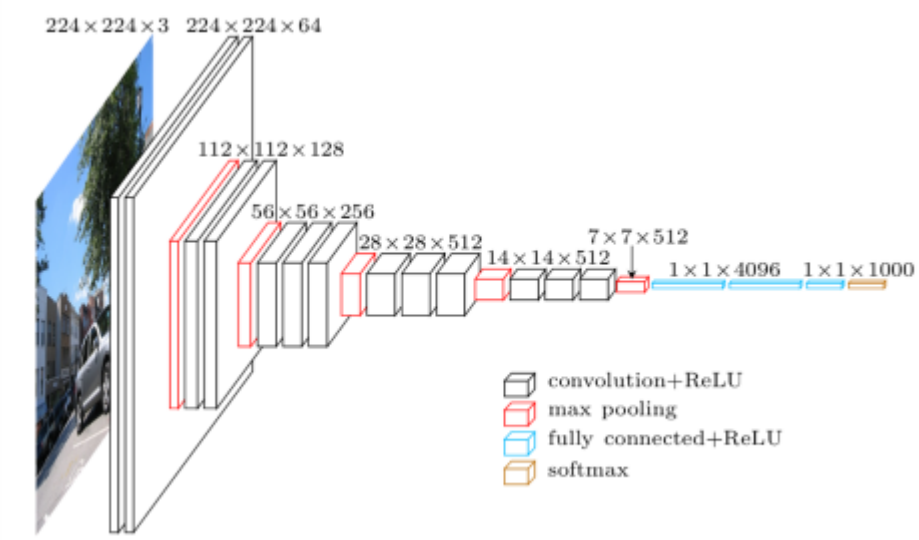
Data preprocessing

All the images are resized to 256 by 256 and then cropped to 224 by 224, which is the standard resolution for VGG16 input. Then all the images are normalized.
In addition, images from training dataset are augmented by flipping and rotating them. This image augmentation help our model to generalize better.
In order to send images to the model all of them are also converted into tensor.

Implementation

Image classifier from scratch

When building a model from scratch, I took inspiration from VGG16 architecture. It's quite elegant, compact and easy to understand compared to others, much larger models like ResNet or Inception, and yet very effective.



VGG16 architecture.

My model also take a 224 by 224 image and then in five steps produces 512 feature maps of 7x7. Every convolutional layer doubles the number of feature maps and every pooling layer decreases by two their size.
These 512 feature maps are flattened and sent to the classifier.
In this case, classifier has two fully connected layers where the last one has 133 outputs – from the number of dog breeds. Dropout layers make sure the model do not overfit.

Image classifier with transfer learning

I decided to stick with the VGG16 architecture when building a transfer learning classifier. I used the feature extractor part from pre-trained VGG16 model but replaced the classifier part with my dog breed classifier. For that, I took the same classifier that I used earlier in the model build from scratch.

Refinement

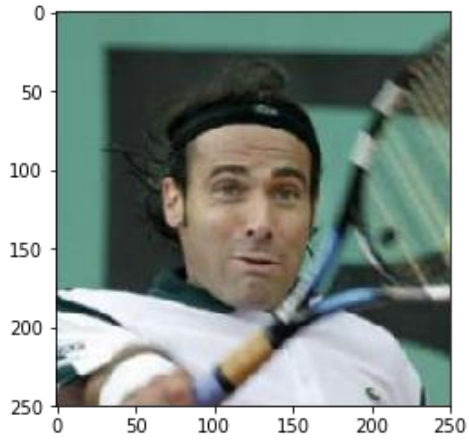
Both model, especially the one built from scratch, needed some tuning to get to the results. I needed to add more layers and modify their dimensions. I also changed such parameters as learning rate and batch size. Bigger networks with more layers tend to perform better but take more time to train and prone to overfitting. To tackle overfitting I used image augmentation and added dropout layers after fully connected layers.

4. Results

Both models met their targets.
The final model built on pre-trained VGG16 model demonstrated 74% accuracy, which is good compared to 60% benchmark. Especially considering only 5 epochs of training. Clearly, VGG16 pre-trained on ImageNet database is very good match for this problem. Its feature extraction part already knows all features of dog images, so all we need to do is just change the classifier part with the one specific to our task and train it for only 5 epochs.
The model built from scratch showed 21% accuracy, with 10% benchmark.
It took more training to get this result because the model needed to learn all the features of dog images. However, 20 epochs is not that much compared to how long it take to train a bigger state-of-the-art model like VGG or Inception. This shows the strength of the CNN architecture.

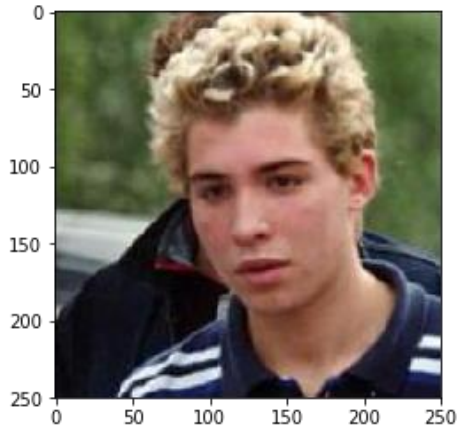
Here are some visual results from the final model.
Images from human dataset first first:

```
hello, human!
```

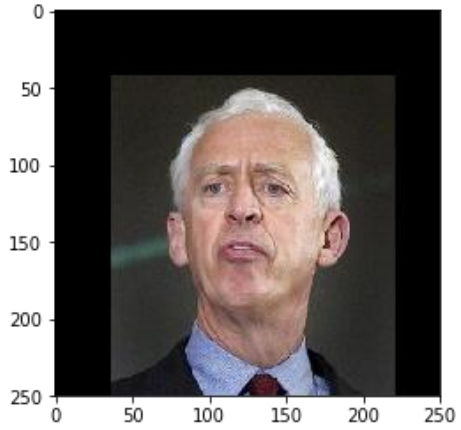



```
You look like a ... Italian greyhound
```

```
hello, human!
```



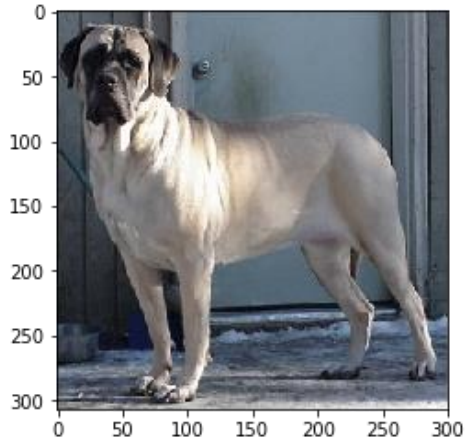
```
You look like a ... Chinese crested

hello, human!
```



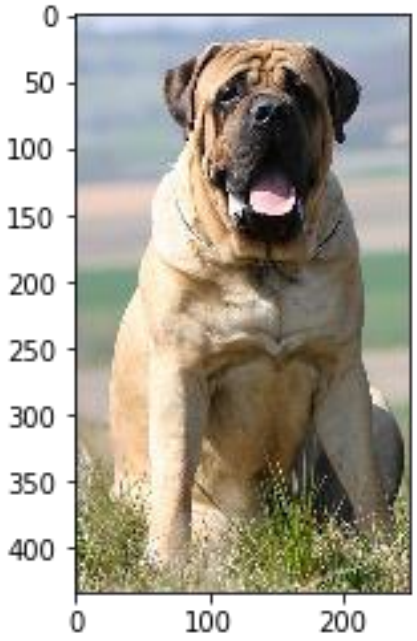
```
You look like a ... Pharaoh hound
```

And here are some results with images from dog dataset:

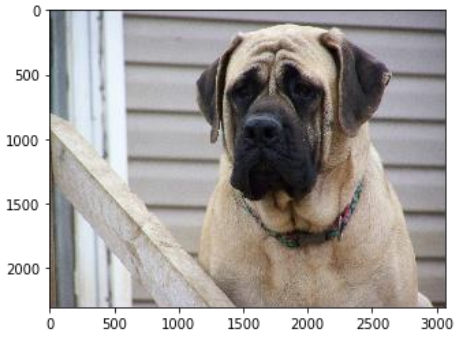
```
This is dog of a following breed: ... Mastiff
```



```
This is dog of a following breed: ... Mastiff
```




```
This is dog of a following breed: ... Mastiff
```



5. References

https://github.com/udacity/dog-project

https://en.wikipedia.org/wiki/Convolutional_neural_network

https://pytorch.org/docs/stable/index.html

https://www.kaggle.com/c/dog-breed-identification/overview/description

http://rajatvikramsingh.github.io/media/DeepLearning_ImageNetWinners.pdf

https://neurohive.io/en/popular-networks/vgg16/