Machine Learning Engineer Nanodegree Capstone Proposal

Cats vs. Dogs
(from a recent Kaggle competition)

Resmi Arjunanpillai
Sep 27, 2017

Domain Background

Computer vision as a space has seen rapid advances in just the last few years. Deep learning has created very high accuracy in recognizing images, leading to applications in multiple areas like health, self driving cars, robotics etc.

A few of the key papers in the space:

- Alexnet is the paper that jump started the current interest in CNNs. This paper by Geoffrey Hinton and others, describes a deep convolutional neural network that won the 2012 Imagenet competition.
  https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
- VGGnet was another influential paper that demonstrated the use of smaller 3x3 filters and promoted deeper networks.
  https://arxiv.org/pdf/1409.1556v6.pdf

Considerable research has also gone into optimizing hyper parameters. A few examples

- Momentum: https://arxiv.org/pdf/1412.6980.pdf and .
- Adam optimizer: https://arxiv.org/pdf/1412.6980.pdf

Problem Statement

https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition
The Dogs vs. Cats Redux: Kernels Edition on Kaggle is a competition to differentiate images of cats from images of dogs.

Though the problem is a well researched one, I think trying out many different approaches to achieve a very high level of accuracy to solve the problem can be challenging. Given the existence of several networks trained on Imagenet within Keras (VGG, Resnet etc), I would like to use transfer learning using pre-trained networks. Optimizing the learning parameters, using batch normalization, data augmentation etc would be important.

Datasets and Inputs

Data: https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition/data

The train folder contains 25,000 images of dogs and cats. Each image in this folder has the label as part of the filename. The test folder contains 12,500 images, named according to a numeric id. For each image in the test set, the goal is to predict a probability that the image is a dog (1 = dog, 0 = cat).

I will split the training set into train/validation groups to make it easier to iterate on my model and optimize it.

There are 12,500 cat images and 12,500 dog images. The naming follows the convention cat.number.jpg and dog.number.jpg. For the validation set, I will use approx 1,000 cat and dog images. The images are full color images with 3 channels - RGB.

The classes are balanced well and I'll maintain a similar class balance in the validation set.

Image dimensions range from 300-500 x 300-500. There are some outliers as well, but the majority of the images are within this range.

Pre-processing needed is the same for Vgg16 and Resnet50 -- remove vgg mean or resnet mean and convert RGB to BGR. Images need to be resized to 224 x 224.

Solution Statement

My goal would be to achieve a logloss of 0.1 or lower on the test set (which will put me in the top 1/3rd of the competitors).

I will start with pre-trained networks and modify them to achieve higher performance.

Benchmark Model

I'll train a VGG16 model on the data and use it as the benchmark model. (Additional details in the final section)

Evaluation Metrics

The contest uses log-loss as the evaluation metric

$$\text{LogLoss} = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)],$$

where

- n is the number of images in the test set

- ŷ iy^i is the predicted probability of the image being a dog
- yiyi is 1 if the image is a dog, 0 if cat
- log()log() is the natural (base e) logarithm

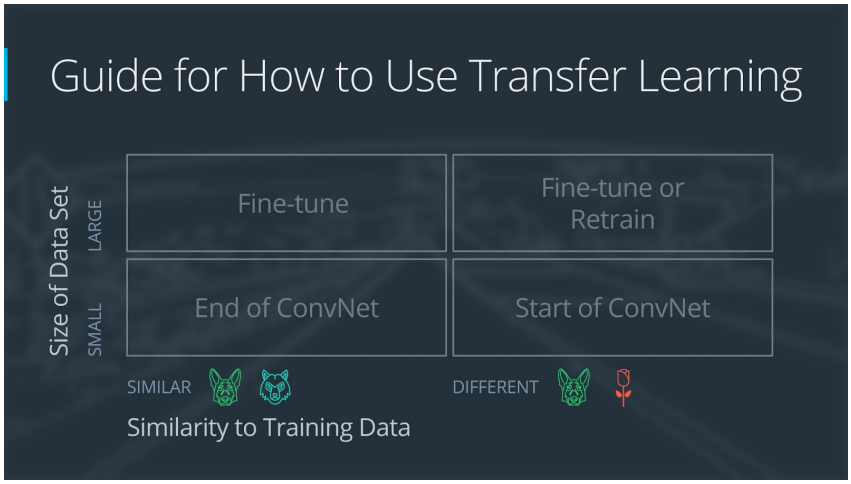
A smaller log loss is better.

LogLoss is related to cross-entropy and measures the performance of a classification model where the prediction is a probability value between 0 and 1. LogLoss takes into account the uncertainty of the prediction based on how much it varies from the actual model.

Project Design

I plan to use 2 pre-trained networks -- Vgg16 and Resnet50 -- in Keras.

The cats and dogs images are similar to the ImageNet images that these networks have been trained on. The cats/dogs dataset is large, but not huge.



Given these, I'll start by replacing the final fully connected layer with a layer with 2 classes. I'll train the final layer while keeping the ImageNet weights for the rest of the network.

The images will need to be pre-processed for each pretrained model. Pre-processing needed is the same for Vgg16 and Resnet50 -- remove vgg mean or resnet mean and convert RGB to BGR. Images need to be resized to 224 x 224.

With Vgg16, I plan to do a few more modifications to try and improve the accuracy/reduce log loss.

1. Re-train additional dense layers - not just the last layer. We have a large enough dataset to do that without considerable overfitting.
2. Change levels of dropout
3. Try different methods of data augmentation.

2 and 3 above should help reduce any overfitting issues.

Picking the right learning rate[1], dropout levels and data augmentation will be key to try and improve the accuracy.

[1] http://cs231n.github.io/transfer-learning/