Machine learning Nanodegree
Capstone proposal

Richard Deurwaarder

November 12th 2016

1 Background

This project is a Kaggle playground competition, called Dogs vs. Cats Redux: Kernels Edition[1]. Originally this competition was run in 2013. With deep learning being more popular these days they have reintroduced the competition. It makes for a nice view into the progression of Machine learning over the last tree years.

2 The Problem

The competition is about determining whether images are either cat or a dog by giving a probability to each image, 0 being cat, 1 being dog. The dataset is called the Asirra[2] data set. It consists of a training set with 25,000 images, half cats and the other half of dogs. The test set has 12,500 images.

3 Solution workflow

3.1 Preprocessing the data

I will need to preprocess the images because they're not all the same size. Creating a frequency table of the sizes of the images in the training set will guide me to the final image size my model will take as its inpu. I will pad the smaller images with grey pixels and crop the bigger images. I might also grayscale the images, depending on how it will effect performance. Intuitively the color should not make much of a difference when determining

[1] `https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition`

[2] Asirra: Animal Species Image Recognition for Restricting Access

if it's a cat or a dog, so it's a good way to reduce the complexity of the model, but it's something I need to test.

3.2 Model

After having cleaned up the dataset I want to try atleast two different but similar approaches. The first one would be using Inception[3] a model in the tensorflow framework. This has pretrained weights for image recognition. It uses a convolutional neural network. The other approach would be to tweak my own convolution neural network (using tensorflow again), and see how to performs versus the inception model. Using a validation set I will determine which one of the model I will end up using.

3.3 Evaluation

I'll use the following as loss function:

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

This is the same loss funcion that Kaggle uses to evaluate the submissions of participants. You can read the logloss function as following, it averages the error for each classification (the $-\frac{1}{n}\sum_{i=1}^{n}$ part). Inside of the sum is the error for each classification. It has two parts, each part calculating the error for when the images was a dog/cat and how our guess corresponds with that. For instance, if the image was a cat and we very confidently guess it's a dog, $y_i = 0$, $\hat{y}_i = 0.999$. The part inside of the sum would become:

$$y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i) = 0 * log(0.999) + 1 * log(0.001) \approx -6.9$$

After getting negated by the $-\frac{1}{n}$ this gives a very high logloss. In other words when being very confidently wrong, the logloss will punish you very hard for it.

4 Compare my model

The competition already has some other submissions (200 at the time of writing) this will allow me to see what algorithms other people have used. In particular I'd like to test it versus the winning submission of 2013, it

[3] `https://github.com/tensorflow/models/tree/master/inception`

had 98,914% correct classifications. Because the competition from 2013 is very similar, it is slightly different. For instance instead of probabilities participants had either pick a dog or cat, 0 or 1. For this reason I would also like to pick one of the submissions the current competition, the highest score so far is a LogLoss of 0.04220.