Machine Learning Engineer Nanodegree

Capstone Proposal

Eddy Shyu August 10, 2017

Proposal

Domain Background

I've chosen a 3D image detection problem that identifies the likelihood and location of threats hidden on a person's body during airport security screening. This is the "Passenger Screening Algorithm Challenge" that is hosted by Kaggle.

I did some search on imaging related to 3D data. Some apply 3D convolutions, but one paper I read suggested "multi-view convolutional neural networks."

The idea of multi-view CNN is to look at the data as a series of 2D slices at different angles around the object, as if one is taking a picture of the object while moving around it in a circle. We can pass the images through convolutional layers that share the same weights for all angles of the image data.

Then we can combine all the angles with a max function, so we end up a tensor that has the same dimensions as a single 2D slice. The rest of the model design looks pretty similar to a typical convolutional neural network with 2D image data.

Problem Statement

The problem is to take 3D scan data of people, and labels that represent whether each of 17 body regions (such as left thigh, right forearm, front torso, etc.) has an object that is a potential threat (an object that isn't normally part of the human body). We want to predict the probability that each region of a new passenger image contains one of these potential threats.

The measure of success is the average log loss function on a test sample. Based on other Kaggle competitor results, I am trying to get below a test loss of 0.29, as many competitors are in the 0.2900 to 0.2999 range, with a rank ranging from 46 to 107 (as of August 10, 2017).

Datasets and Inputs

There are multiple versions of data for the same sample of passengers, with increasing data size per image.

The smallest data set is ".aps" format, which consists of 16 2D slices at various angles around each person (10mb per file)

The next larger data set ".a3daps" consists of 64 2D slices at various angles around each person

(40mb per file)

The next larger data set ".a3d" consists of volume data with 3 coordinates: height, depth, horizontal (330mb per file)

These are available from the Kaggle site, which links to a google cloud platform bucket, which contains the data.

A csv file "stage_1_labels.csv" contains a unique string for each person's image, followed by one of 17 zones that represent locations of the human body, and the label 1 if a threat exists on that person's body for that zone, 0 otherwise.

There are 1147 training samples, and 100 test samples that are used for the Kaggle leaderboard rankings. So there are a total of 1247 files for each data type (either .aps, .a3daps, or .a3d).

Solution Statement

I will transfer learning with a pre-trained VGG network, so I will resize and pad the original images to fit the VGG network. Since the data starts with 1 channel (it's like a gray-scale x-ray image), I will duplicate the single channel for all R,G,B channels, it so that the VGG network can read it.

Then I will try various convolutional layers of the pre-trained network to use as inputs to a trainable multi-view convolutional neural network. The multi-view cnn will re-use weights while looping through all available angles (16 angles for the smallest .aps data set, for instance). Then it will use a max function to combine all the tensors of the various angles into a single tensor. This will pass through some fully connected dense layers and then output 17 logits (one logit for each zone on the body), and finally converted to a probability with a sigmoid.

The network will optimize over a log loss function.

Benchmark Model

My benchmark will be the Kaggle leaderboards; I want to get in the top 80 of the 164 participants. This requires a test loss of less than 0.29098.

Evaluation Metrics

The model can be evaluated by how well its predictions align with the actual labels of the final test data set. This is the average log loss, which is the actual label times the log of the predicted probability. Log loss is essentially cross-entropy when there are only 2 classes, When there are only 2 classes, we don't have to one-hot encode the label, as a single digit can represent the two classes as either 0 or 1.

Project Design

Exploratory Data Analysis

- I will use EDA to better understand the image dimensions and how the object looks at various

angles.

Data Pre-processing

- I will pad, resize, and rescale the image. I will also replicate the 1 channel into 3 channels, so that the image matches the required RGB input of the pre-trained VGG network.
- I will also normalize to range bewteen 0 and 1, which is the range expected by the vgg network.

Model Design

- I will start with adding 3 additional convolutional layers, and 2 fully connected dense layers.
- I will use batch normalization, leaky relus, and dropout for each layer. I will use max pooling for each convolutional layer.
- I will try using the outputs of various layers from the pre-trained network, for both vgg16 and the vgg19 model.
- I will also try using residual layers and bottlenecks for the trainable layers.
- I may try a different pre-trained network. For instance, google offers an inception-resnet model. It takes a lot of extra steps to set up, so I am starting with vgg first.
- After I've tried various designs and can't make any more improvements using the smallest data set (.aps), I can try the larger data set with 64 angles instead of 16.