Machine Learning Advanced Nanodegree

Capstone Proposal for Classifier for Kaggle's Amazon from Space competition

Vivek Pothina

December 21st, 2018

Domain Background

A few decades ago, Image Classification was a difficult task for computers, but now using Deep Learning we can empower satellites to label satellite images. Planet: Understanding Amazon from Space is competition in Kaggle. Planet, designer and builder of the world's largest constellation of Earth-imaging satellites, collected images of 3-5-meter resolution around Amazon Rainforest. On Kaggle, the competition was a challenge to label satellite image chips with atmospheric conditions and various classes of land cover/land use. This is a multi-label classification problem too. Using a simple neural network to solve this problem would be slow and ineffective. CNN/Convnet has been used to use the spatial information to capture features of the satellite images such as cloudy or rain forest features.

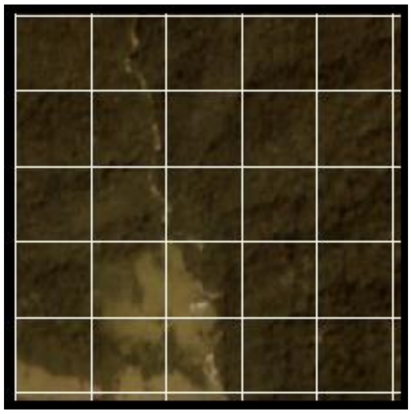
Problem Statement

The input is a satellite image chip, we need to determine the classes under which the image has a high probability.

Dataset and Inputs

The data was obtained from Kaggle. The data consists of train-jpg.tar.7z and train.csv, the dataset consists of around 40479 images which belong one or more classes among 17 classes. The images are in jpg format, each of shape (256, 256, 4). The color space is RGB + near infrared band. Each image has a unique id and the labels are given in train.csv file. The validation set has been created with a split of 80% from training set.

```
Label: agriculture clear habitation primary road
Shape: (256, 256, 4)
Max: 255, Min: 0
```



Solution Statement

As mentioned above, a CNN was used to solve the problem. The CNN will learn the important features and classify the image to one or more classes. The loss function used was the binary cross entropy function which is present in Keras.

$$-\frac{1}{N}\sum_{n=1}^{N}[y_n \log(\widetilde{y_n}) + (1 - y_n) \log(1 - \widetilde{y_n})]$$

Where N = 17 i.e. the number of classes/labels in the dataset, y_n is the label vector and $\widetilde{y_n}$ is the prediction vector. Weights and biases are updated using the Adam optimizer using a learning rate of 0.001

Benchmark Model

One kaggler achieved accuracy of 93% on the validation set.
https://gist.github.com/EKami/33ec0172590ab9f2e3a6b757c9f9dcb4

The aim of this project is to try to achieve at least 70%-80% using my model.

Evaluation Metrics

As mentioned above, the evaluation metric would be a multi-label log loss function between the predicted probabilities and the actual value for the image.

Project Design

The steps taken to complete this project are:

1. Import the labels from the .csv file
2. Resize all the images to 128, 128. Color data of the images has been considered.
3. The CNN architecture used in the project is:
    - Conv 1: inputs = 32, kernel_size = (3,3)
    - Maxpool Layer
    - Conv 2: inputs = 64, kernel_size = (3,3)
    - Maxpool Layer
    - Conv 3: inputs = 128, kernel_size = (3,3)
    - Dropout: probability = 0.7
    - Flatten
    - Dense or fully connected layer: inputs = 512
    - Output layer with sigmoid activation function
4. For this project, Google Colab notebooks (https://colab.research.google.com) has been used to use a cloud GPU for better machine efficiency.
5. Test the network's performance and tweak the performance.