Machine Learning Engineer Nanodegree

Capstone Proposal

Hoang Chung Hien
February 14th, 2018

Proposal

Domain Background

While go through vacation photos, people always ask themself: What is the name of this temple I visited in Thailand? Who created this monument in France? Landmark recognition can help! This technology can predict landmark labels directly from image pixels, to help people better understand and organize their photo collections.

In this project, a model use for landmark recognition will be built based on ImageNet Classification With Deep Convolutional Neural Networks [1]

Problem Statement

The goal of this project is to build a classification model that will be used to recognize landmark from a photo. A few problems need to be solved to achieve the goal:

- The landmark dataset that will be used to train the model, this can be solved by using dataset provided by Google Landmark Recognition Challenge [2].
- Train a classifier model using the dataset above would take a lot of time and computational resource. This problem can be solved by using transfer learning [3].

Datasets and Inputs

The training set was constructed by clustering photos with respect to their geolocation and visual similarity using an algorithm similar to the one described in [4]. Matches between training images were established using local feature matching. Note that there may be multiple clusters per landmark, which typically correspond to different views or different parts of the landmark.

The dataset provided by [5] which is obtained from a kaggle challenge [2]. The provided dataset was very large and could contains up to 300 gigabytes of images which belongs to 14951 classes. Download all the images will take a week, train for only one model will take a month (based on my current limiting computational resource). For that reason, I will use only images

---

[1] http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
[2] https://www.kaggle.com/c/landmark-recognition-challenge/data
[3] https://en.wikipedia.org/wiki/Transfer_learning
[4] Y.-T. Zheng, M. Zhao, Y. Song, H. Adam, U. Buddemeier, A. Bissacco, F. Brucher T.-S. Chua, and H. Neven, “Tour the World: Building a Web-Scale Landmark Recognition Engine,” Proc. CVPR 09
[5] H. Noh, A. Araujo, J. Sim, T. Weyand, B. Han, "Large-Scale Image Retrieval with Attentive Deep Local Features", Proc. ICCV'17

from 100 classes and split the training dataset into 75% for training, 25% for validation and testing.

Solution Statement

In this project, I will use a VGG16 pre-trained model [6] implemented by keras team [7]. This is the Keras [8] model of 16-layer network used by VGG team in the ILSVRC-2014 competition [9]. This pre-trained model would do a hardest task of extracting features from an image and act as input for my fully connected deep neural network to detect which landmark that image belongs to.

Benchmark Model

To create benchmark for the solution, I will build a simple CNN model and train with the same dataset that I will use to train my solution model. To understand how best my solution perform, I compare the accuracy score in both training and validation of the solution model with the simple one.

Evaluation Metrics

This is a categorical classifier model which can be evaluate effectively using categorical accuracy metric.

$$accuracy = \frac{number\ of\ correct\ classes\ predicted}{dataset\ size}$$

Project Design

The workflow that I will use to approach the solution require the following steps:

1. Select 100 classes which have highest number of images to download by analyze the train.csv file.
2. Clean the corrupted images. Split the dataset downloaded into 75% for training, 25% for validation and testing.
3. Define a simple CNN using keras. The simple CNN contains 3 convolutional layers with 32, 32, 64 nodes respectively. The last convolutional layer connect with a fully connected network which contains 2 layers with 512, 100 (classes size) nodes respectively.
4. Train the simple CNN for about 100 epochs with early stopping if the accuracy score doesn't increase in few epochs. Save the score for benchmark.
5. Define solution network using VGG16 model to extract features and use as input for a fully connected network which contains 2 layers with 512, 100 (classes size) nodes respectively.
6. Train the solution model for about 100 epochs with early stopping if the accuracy score doesn't increase in few epochs. Save the score for benchmark.
7. Plot the score collected from training process of all model and compare the performance.

[6] Very Deep Convolutional Networks for Large-Scale Image Recognition K. Simonyan, A. Zisserman arXiv:1409.1556
[7] https://github.com/keras-team/keras/blob/master/keras/applications/vgg16.py
[8] https://keras.io/
[9] http://www.image-net.org/challenges/LSVRC/2014/