Machine Learning Engineer Nanodegree

Capstone Proposal

Zhikun Lu
Sep 1st, 2021

Proposal - Dog Breed Classifier

Domain Background

With the recent advancement in computer vision, machine learning technologies have been widely used in real world to solve real problems. A well-known example is the Face ID developed by Apple, used by tens of millions of people everyday. At the core of a technology like this is deep learning, or more specifically, deep convolutional neural networks (CNNs), which are extremely powerful for image recognition tasks.

Pioneered by Yann LeCun et al. (1989), CNNs are able to recognize image patterns utilizing its unique architecture. The power of CNNs is then fully awakened by GPU computing (Chellapilla et al, 2006), where training a large-scale CNN became feasible. Nowadays, well-trained CNNs can achieve super-human performance in many image recognition tasks.

In this project, I will build a pipeline to process real-world, user-supplied images, and then train CNNs using these image data with the help of GPU computing. The fully trained model is expected to achieve near-human performance in dog breed classification.

Problem Statement

The central goal of this project is to design and implement a machine-learning-powered application to classify a dog's breed when given a photo. For example, when a photo of dog (see the following picture) is presented, the application is expected to give its correct breed, Labrador Retriever.

To achieve this goal, I will

1. build and train a CNN from scratch, and

2. use a transfer learning model.

Datasets and Inputs

As the goal is to train a deep learning model to classify a bog's breed, the main inputs are images of dog and labels.

- Dog image dataset (main). It can be downloaded from here.
  There are 8351 total dog images, where 6680 of them are used for training, 836 for testing, and 835 for validation. Further, there are 133 different dog breeds.
  The sample is not completely balanced. In the training set, the breed with largest size has 77 images, while the one with smallest size has only 26 images. This is natural as the data is collected from real world.

Occasionally, the project uses human images for comparison and/or other goals.

- Human image dataset (supplementary). It can be downloaded from here.
  There are 13233 total human images.
  We do not use human images to train our CNNs.

All datasets for this project are provided by Udacity.

Solution Statement

I will mainly use CNN to solve our central image classification problem. CNN is a class of artificial neural network that have been widely used for image processing and analysis. Its unique network architecture is able to capure two-dimensional features in pictures and then use these features to draw conclusions. CNN is believed to be one of the most powerful machine/deep learning technologies for image processing.

Benchmark Model

- A CNN model built from scratch. As a starting point, I will design and train a small scale CNN model using my laptop. A toy model should work to some extent but may not be powerful enough to achieve high performance.
  I would set the benchmark accuracy rate to be 10%. Though it seems pretty low, this level is way higher than a random guess, the accuracy rate of which is approximately 0.75% (1/133)
- A CNN model with transfer learning. Transfer learning allows our model to use "knowledge" from other trained CNN models to serve our goal. Those trained CNN models are usually large in scale and powerful in performance, but not tailored for our task at hand. By utilizing their "knowledge" and train a transfer learning model, one can potentially achieve high model performance very fast.
  I would set the benchmark accuracy rate for a tranfered model to be 80%. As a person who can barely differentiate one breed from another, I believe this is a pretty high standard.

Evaluation Metrics

For model training, I use cross-entropy loss, which is standard choice for multi-class classification problems.

Project Design

I will mainly take the following steps for this project:

1. Load and pre-process the image data
   To expand effective sample size and to reduce overfitting, I'll resize, rotate, flip and crop images in a random manner.
2. Train a CNN model from Scratch
   I will fisrt try a CNN with three convolutional layers. If it fails to 10% goal, I will think about expand the neutral network size.
3. Train a CNN model with transfer learning
   I will fisrt try VGG16 for the transfer learning. Alternative pre-trained models will be employed if the 80% goal cannot be achived with VGG16.
4. Test and evaluate the model

Reference

1. LeCun, Yann, et al. "Backpropagation applied to handwritten zip code recognition." Neural computation 1.4 (1989): 541-551.
2. Chellapilla, Kumar, Sidd Puri, and Patrice Simard. "High performance convolutional neural networks for document processing." Tenth international workshop on frontiers in handwriting recognition. Suvisoft, 2006.
3. CNN Project: Dog Breed Classifier, Udacity's github repository https://github.com/udacity/deep-learning-v2-pytorch/tree/master/project-dog-classification
4. The Dog image dataset https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/dogImages.zip
5. The human image dataset http://vis-www.cs.umass.edu/lfw/lfw.tgz
6. PyTorch Document https://pytorch.org/docs/master/generated/torch.nn.CrossEntropyLoss.html