Caffe

Deep learning framework by the BVLC

Created by
Yangqing Jia
Lead Developer
Evan Shelhamer

View On GitHub

Classifying ImageNet: using the C++ API

Caffe, at its core, is written in C++. It is possible to use the C++ API of Caffe to implement an image classification application similar to the Python code presented in one of the Notebook example. To look at a more general-purpose example of the Caffe C++ API, you should study the source code of the command line tool `caffe` in `tools/caffe.cpp`.

Presentation

A simple C++ code is proposed in `examples/cpp_classification/classification.cpp`. For the sake of simplicity, this example does not support oversampling of a single sample nor batching of multiple independant samples. This example is not trying to reach the maximum possible classification throughput on a system, but special care was given to avoid unnecessary pessimization while keeping the code readable.

Compiling

The C++ example is built automatically when compiling Caffe. To compile Caffe you should follow the documented instructions. The classification example will be built as `examples/classification.bin` in your build directory.

Usage

To use the pre-trained CaffeNet model with the classification example, you need to download it from the "Model Zoo" using the following script: `./scripts/download_model_binary.py models/bvlc_reference_caffenet` The ImageNet labels file (also called the synset file) is also required in order to map a prediction to the name of the class: `./data/ilsvrc12/get_ilsvrc_aux.sh`. Using the files that were downloaded, we can classify the provided cat image (`examples/images/cat.jpg`) using this command: `./build/examples/cpp_classification/classification.bin \ models/bvlc_reference_caffenet/deploy.prototxt \ models/bvlc_reference_caffenet/bvlc_reference_caffenet.caffemodel \ data/ilsvrc12/imagenet_mean.binaryproto \ data/ilsvrc12/synset_words.txt \ examples/images/cat.jpg` The output should look like this: `---------- Prediction for examples/images/cat.jpg ---------- 0.3134 - "n02123045 tabby, tabby cat" 0.2380 - "n02123159 tiger cat" 0.1235 - "n02124075 Egyptian cat" 0.1003 - "n02119022 red fox, Vulpes vulpes" 0.0715 - "n02127052 lynx, catamount"`

Improving Performance

To further improve performance, you will need to leverage the GPU more, here are some guidelines:

- Move the data on the GPU early and perform all preprocessing operations there.
- If you have many images to classify simultaneously, you should use batching (independent images are classified in a single forward pass).
- Use multiple classification threads to ensure the GPU is always fully utilized and not waiting for an I/O blocked CPU thread.