Caffe

Deep learning framework by BAIR

Created by
Yangqing Jia
Lead Developer
Evan Shelhamer

View On GitHub

Extracting Features

In this tutorial, we will extract features using a pre-trained model with the included C++ utility. Note that we recommend using the Python interface for this task, as for example in the filter visualization example.

Follow instructions for installing Caffe and run scripts/download_model_binary.py models/bvlc_reference_caffenet from caffe root directory. If you need detailed information about the tools below, please consult their source code, in which additional documentation is usually provided.

Select data to run on

We'll make a temporary folder to store things into.

```
mkdir examples/_temp
```

Generate a list of the files to process. We're going to use the images that ship with caffe.

```
find `pwd`/examples/images -type f -exec echo {} \; > examples/_temp/temp.txt
```

The ImageDataLayer we'll use expects labels after each filenames, so let's add a 0 to the end of each line

```
sed "s/$/ 0/" examples/_temp/temp.txt > examples/_temp/file_list.txt
```

Define the Feature Extraction Network Architecture

In practice, subtracting the mean image from a dataset significantly improves classification accuracies. Download the mean image of the ILSVRC dataset.

```
./data/ilsvrc12/get_ilsvrc_aux.sh
```

We will use data/ilsvrc212/imagenet_mean.binaryproto in the network definition prototxt.

Let's copy and modify the network definition. We'll be using the ImageDataLayer, which will load and resize images for us.

```
cp examples/feature_extraction/imagenet_val.prototxt examples/_temp
```

Extract Features

Now everything necessary is in place.

```
./build/tools/extract_features.bin models/bvlc_reference_caffenet/bvlc_reference_caffenet.caffemodel examples/_temp/imagenet_val.prototxt fc7 examples/_temp/features
10 leveldb
```

The name of feature blob that you extract is `fc7`, which represents the highest level feature of the reference model. We can use any other layer, as well, such as `conv5` or `pool3`.

The last parameter above is the number of data mini-batches.

The features are stored to LevelDB `examples/_temp/features`, ready for access by some other code.

If you meet with the error “Check failed: status.ok() Failed to open leveldb examples/_temp/features”, it is because the directory examples/_temp/features has been created the last time you run the command. Remove it and run again.

```
rm -rf examples/_temp/features/
```

If you’d like to use the Python wrapper for extracting features, check out the filter visualization notebook.

Clean Up

Let’s remove the temporary directory now.

```
rm -r examples/_temp
```