Caffe: Convoluational Architecture for Fast Feature Embedding

Chang Liu

April 3, 2015

Contents

1 Abstract

This section gives a basic idea about Caffe, which is as follows that matter:

1) C++ libarary with Python and Matlab binding for training and deploying

2) General convolutional neural network and **other deep-learning models**

3) CUDA GPU computing, with 40 million images a day on single K40 or Titan GPU

2 Introduction

Selction:

1) hand-craft feature is not as good as learning features in representing images.

2) It usually takes months to replicate the result of others' research, which makes it hard for improving.

3) By designing Caffe, it gives a very good result and application in research and industry.

Framework	License	Core language	Binding(s)	CPU	GPU	Open source	Training	Pretrained models	Development
Caffe	BSD	C++	Python, MATLAB	✓	✓	✓	✓	✓	distributed
cuda-convnet [7]	unspecified	C++	Python		✓	✓	✓		discontinued
Decaf [2]	BSD	Python		✓		✓	✓	✓	discontinued
OverFeat [9]	unspecified	Lua	C++,Python	✓				✓	centralized
Theano/Pylearn2 [4]	BSD	Python		✓	✓	✓	✓		distributed
Torch7 [1]	BSD	Lua		✓	✓	✓	✓		distributed

Table 1: Comparison of popular deep learning frameworks. *Core language* is the main library language, whi *bindings* have an officially supported library interface for feature extraction, training, etc. *CPU* indicat availability of host-only computation, no GPU usage (e.g., for cluster deployment); *GPU* indicates the GP computation capability essential for training modern CNNs.

3 Highlights

these advantages:

1) Modularity. easy extension to new data.

2) Separation of representation and implementation. model defintion using Proftocol BUffer.

3) Test coverage.

4) Python and MATLAB binding.

5) Pre-trained reference models. AlexNet, ImageNet, R-CNN models

3.1 Comparing to other software

These contributions make the Caffe outstanding to other software:

1) Caffe is purely C++ based, making it easy to integration to existing C++ system.

2) Reference model is provided for quick experiment, without need for costly re-learning.

4 Architecture

This section gives a very good description about Caffe's structure.

4.1 Data storage

Main points:

1). communicates data in 4-dimensional array called blobs.
2). conceal the computational and mental overhead of mixed CPU/GPU.
3). Large scale data is store in LevelDB database, layer-wise design...

4.2 Layers

A couple of layers types, includes, convolution, pooling, inner products, nonlinearities like rectified linear and logistic, local response normalization, element-wise operations, and losses like softmax and hinge

4.3 Network and Run Mode

CPU/GPU mixture.

4.4 Training A network

reduces the loss and gradients which train the whole network. This example is found in the Caffe source code at examples/lenet/lenet_train.prototxt. Data are processed in mini-batches that pass through the network sequentially. Vital to training are learning rate decay schedules, momentum, and snapshots for stopping and resuming, all of which are implemented and documented.

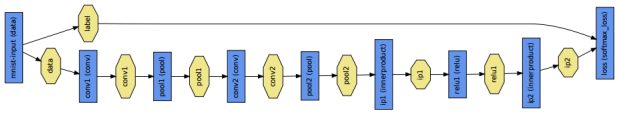

Figure 1: An MNIST digit classification example of a Caffe network, where blue boxes represent layers and yellow octagons represent data blobs produced by or fed into the layers.

4.5 Application and Examples

4.6 Object Classification

models: 10,000 categories of the full ImageNet dataset by finetuning this network

4.7 Learning Semantic Features

4.8 Object Detection

Girshick et al. have combined Caffe together with techniques such as Selective Search to effectively perform simultaneous localization and recognition in natural images