Getting started with Caffe
Jon Barker, Solutions Architect
Agenda

Caffe tour
  Overview
  Example applications
  Setup
  Performance
  Hands-on lab preview
A tour of Caffe
What is Caffe?

An open framework for deep learning developed by the Berkeley Vision and Learning Center (BVLC)

- Pure C++/CUDA architecture
- Command line, Python, MATLAB interfaces
- Fast, well-tested code
- Pre-processing and deployment tools, reference models and examples
- Image data management
- Seamless GPU acceleration
- Large community of contributors to the open-source project

caffe.berkeleyvision.org
http://github.com/BVLC/caffe
What is Caffe?
End-to-end Deep Learning for the practitioner and developer

Prototype
Train
Deploy
## Caffe features

### Data pre-processing and management

<table>
<thead>
<tr>
<th>Data ingest formats</th>
<th>Pre-processing tools</th>
<th>Data transformations</th>
</tr>
</thead>
<tbody>
<tr>
<td>LevelDB or LMDB database</td>
<td>LevelDB/LMDB creation from raw images</td>
<td>Image cropping, resizing, scaling and mirroring</td>
</tr>
<tr>
<td>In-memory (C++ and Python only)</td>
<td>Training and validation set creation with shuffling</td>
<td>Mean subtraction</td>
</tr>
<tr>
<td>HDF5</td>
<td>Mean-image generation</td>
<td></td>
</tr>
<tr>
<td>Image files</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$\text{CAFFE\_ROOT}/\text{build/tools}$
Caffe features
Deep Learning model definition

- Protobuf model format
  - Strongly typed format
  - Human readable
  - Auto-generates and checks Caffe code
  - Developed by Google
  - Used to define network architecture and training parameters
  - No coding required!

```plaintext
name: "conv1"
type: "Convolution"
bottom: "data"
top: "conv1"
convolution_param {
  num_output: 20
  kernel_size: 5
  stride: 1
  weight_filler {
    type: "xavier"
  }
}
```
Caffe features

Deep Learning model definition

- Loss functions:
  - Classification
    - Softmax
    - Hinge loss
  - Linear regression
    - Euclidean loss
  - Attributes/multiclassification
    - Sigmoid cross entropy loss
  - and more...

- Available layer types:
  - Convolution
  - Pooling
  - Normalization

- Activation functions:
  - ReLU
  - Sigmoid
  - Tanh
  - and more...
Caffe features

Deep Neural Network training

Network training also requires no coding - just define a “solver” file

net: “lenet_train.prototxt”
base_lr: 0.01
momentum: 0.9
max_iter: 10000
snapshot_prefix: “lenet_snapshot”
solver_mode: GPU

> caffe train --solver lenet_solver.prototxt --gpu 0

Multiple optimization algorithms available: SGD (+momentum), ADAGRAD, NAG
Caffe features

Monitoring the training process

Output to stdout:

To visualize - pipe, parse and plot or use DIGITS
Caffe features

Deep Neural Network deployment

Standard, compact model format

`caffe train` produces a binary `.caffemodel` file

Easily integrate trained models into data pipelines

Deploy against new data using command line, Python or MATLAB interfaces

Deploy models across HW and OS environments

`.caffemodel` files transfer to any other Caffe installation (including DIGITS)
Caffe features

Deep Neural Network sharing

Caffe Model Zoo hosts community shared models

Benefit from networks that you could not practically train yourself

https://github.com/BVLC/caffe/wiki/Model-Zoo

Caffe comes with unrestricted use of BVLC models:

AlexNet
R-CNN
GoogLeNet

Caffe model directory

Solver + model prototxt(s)

readme.md containing:
- Caffe version
- URL and SHA1 of .caffemodel

License

Description of training data
Caffe features

Extensible code

```python
import caffe
import numpy as np

class EuclideanLoss(caffe.Layer):
    def setup(self, bottom, top):
        # check input pair
        if len(bottom) != 2:
            raise Exception("Need two inputs to compute distance.")

    def reshape(self, bottom, top):
        # check input dimensions match
        if bottom[0].count != bottom[1].count:
            raise Exception("Inputs must have the same dimension.")
        # difference is shape of inputs
        self.diff = np.zeros_like(bottom[0].data, dtype=np.float32)
        # loss output is scalar
        top[0].reshape(1)

    def forward(self, bottom, top):
        self.diff[:] = bottom[0].data - bottom[1].data
top[0].data[...] = np.sum(self.diff**2) / bottom[0].num / 2.

    def backward(self, self, top, propagate_down, bottom):
        for i in range(2):
            if not propagate_down[i]:
                continue
            if i == 0:
                sign = -1
            else:
                sign = -1
            bottom[i].diff[...] = sign * self.diff / bottom[i].num
```

Layer Protocol == Class Interface

Define a class in C++ or Python to extend Layer

Include your new layer in a network prototxt

```protobuf
layer {
  type: "Python"
  python_param {
    module: "layers"
    layer: "EuclideanLoss"
  }
}
```
Caffe example applications
Example applications

Use case 1: classification of images

**Object**

http://demo.caffe.berkeleyvision.org/

Open source demo code:

$CAFFE_ROOT/examples/web_demo

**Scene**

http://places.csail.mit.edu/

B. Zhou et al. NIPS 14

**Style**

http://demo.vislab.berkeleyvision.org/

Karayev et al. *Recognizing Image Style*. BMVC14
Example applications

Use case 2: localization

(Fast) Region based Convolutional Networks (R-CNN)
Ross Girshick, Microsoft Research
https://github.com/rbgirshick/fast-rcnn
Example applications

Use case 3: pixel level classification and segmentation

[Diagram]

http://fcn.berkeleyvision.org

Example applications

Use case 4: Sequence learning

- Recurrent Neural Networks (RNNs) and Long Short Term Memory (LSTM)
  - Video
  - Language
  - Dynamic data

- Current Caffe pull request to add support
  - https://github.com/BVLC/caffe/pull/1873
  - http://arxiv.org/abs/1411.4389

A group of young men playing a game of soccer.

Jeff Donahue et al.
Example applications

Use case 5: Transfer learning

Lots of data

CNN

Transfer weights

New data

ImageNet

© kaggle.com

CNN

Object Classifier

Dog vs. Cat

Top 10 in 10 mins after finetuning

Just change a few lines in the model prototxt file

layer {
  name: “data”
  type: “Data”
  data_param {
    source: “ilsvrc12_train”
    …
  }
  …
}
…

layer {
  name: “fc8”
  type: “InnerProduct”
  inner_product_param {
    num_output: 1000
    …
  }
  …
}
…

layer {
  name: “data”
  type: “Data”
  data_param {
    source: “dogcat_train”
    …
  }
  …
}
…

layer {
  name: “fc8-dogcat”
  type: “InnerProduct”
  inner_product_param {
    num_output: 2
    …
  }
  …
}
…
Caffe setup and performance
Caffe setup

NVIDIA fork enables multiGPU: https://github.com/NVIDIA/caffe

- Tried and tested by BVLC on Ubuntu 14.04/12.04 and OS X 10.8+
- Also demonstrated to compile on RHEL, Fedora and CentOS
- Download source from https://github.com/BVLC/caffe
- Unofficial 64-bit Windows port https://github.com/niuziheng/caffe
- Linux setup (see http://caffe.berkeleyvision.org/installation.html)
  - Download
  - Install pre-requisites
  - Install CUDA and cuDNN for GPU acceleration
  - Compile using make
GPU acceleration

\[-\text{gpu N flag tells caffe which gpu to use}\]

Alternatively, specify solver_mode: GPU in solver.prototxt
cuDNN integration
http://developer.nvidia.com/cudnn

Drop-in support

Install cuDNN, uncomment `USE_CUDNN :=1` in `Makefile.config` before build.
Caffe model mobile deployment

➤ Jetson TX1
   ➤ NVIDIA Maxwell™ GPU with 256 NVIDIA® CUDA® Cores
   ➤ 4 GB LPDDR4 Memory, 16 GB eMMC 5.1 Flash Storage
   ➤ Connects to 802.11ac Wi-Fi and Bluetooth enabled devices 10/100/1000BASE-T
   ➤ No need to change code

➤ Simply compile Caffe and copy a trained .caffemodel to TK1

*Source: http://petewarden.com/2014/10/25/how-to-run-the-caffe-deep-learning-vision-library-on-nvidias-jetson-mobile-gpu-board/
Hands-on lab preview

bit.ly/dlnvlab3

- Use data pre-processing tools
- Edit a network definition
- Train a model
- Improve classification accuracy by modifying network parameters
- Visualize trained network weights
- Deploy a model using Python
Deep Learning Lab Series Schedule
developer.nvidia.com/deep-learning-courses

- Review the other seminars in series

Seminar #2 - Introduction to DIGITs
Seminar #4 - Getting Started with the Theano Framework
Seminar #5 - Getting Started with the Torch Framework
Hands-on Lab

1. Create an account at nvidia.qwiklab.com
2. Go to “Getting started with Caffe” lab at bit.ly/dlnvlab3
3. Start the lab and enjoy!

Only requires a supported browser, no NVIDIA GPU necessary!
Lab is free until end of Deep Learning Lab series