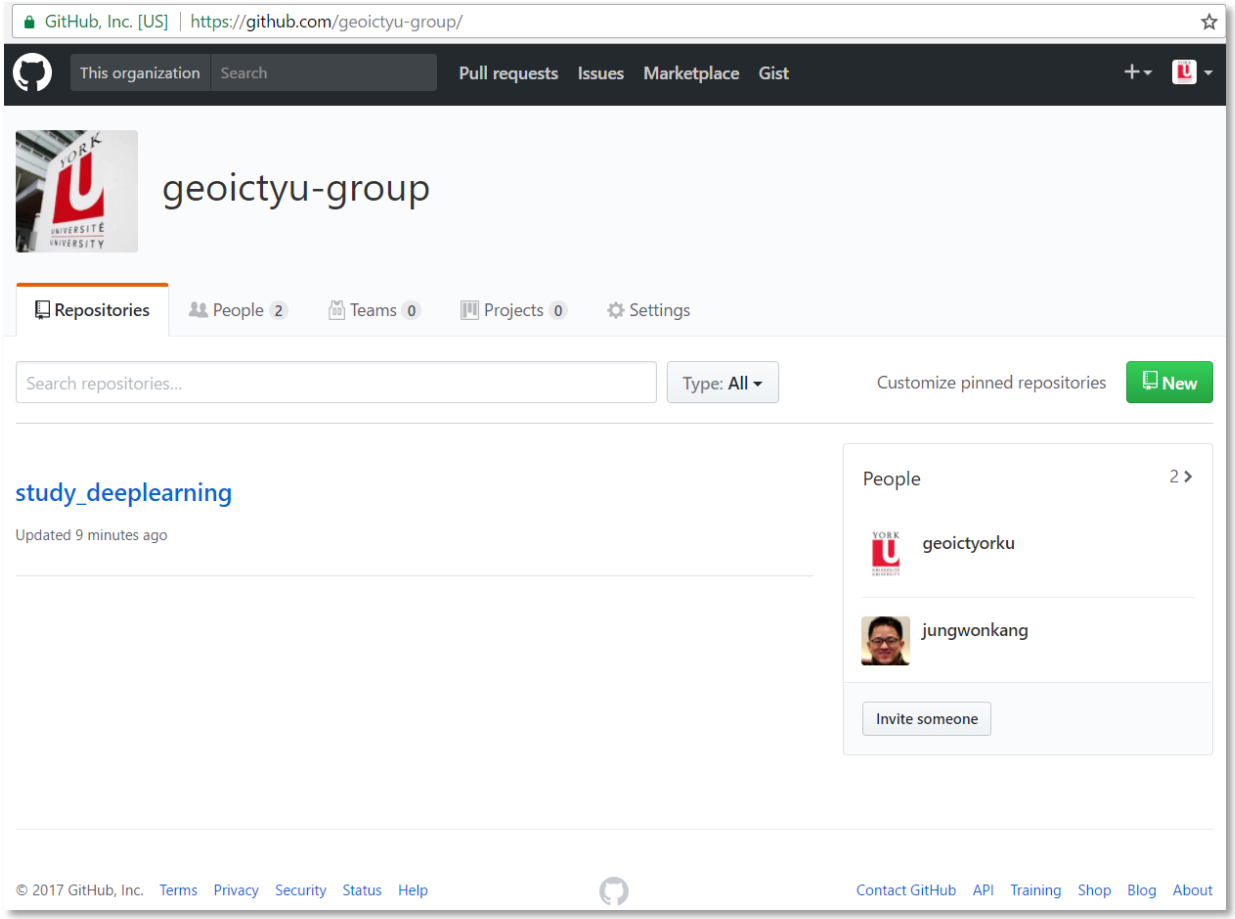
Caffe Programming Guide

June 15 2017

Jungwon Kang

Our Group Site Opens

- https://github.com/geoictyu-group/



Presentation Schedule

Date	Topic
6/1	**Starting Caffe** • Caffe Installation • Caffe Programming Overview
6/15	**Caffe Programming Guide**
TBD	**Object Detection Using Caffe**

Contents

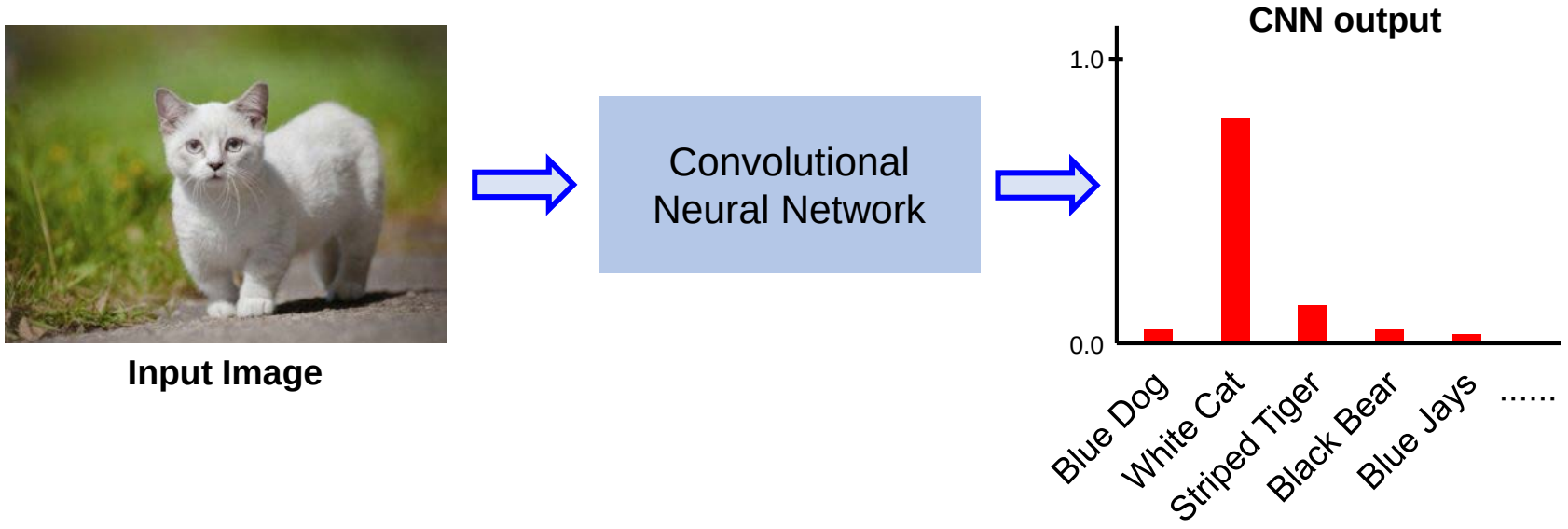
Installation of Caffe

- **Recommendation**
  - NVIDIA Graphic Card (such as NVIDIA TITAN or GeForce Series)
  - Linux: Ubuntu 14.04 or later

- **If not familiar with linux, you can install Caffe on Windows:**
  - [Code] Microsoft/caffe: Caffe on both Linux and Windows
    https://github.com/Microsoft/caffe
  - [Guide] How to install Caffe in windows in 5 min
    https://youtu.be/nrzAF2sxHHM
  - [Guide] How to Build Caffe, Pycaffe, Matcaffe on Windows 10
    https://youtu.be/AN2uXGRvw9E

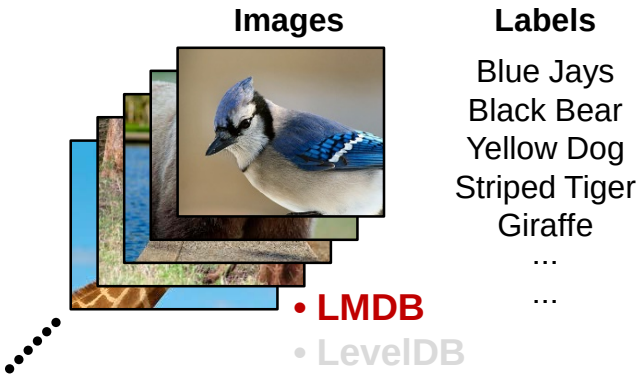
- **Other choices for Windows users:**
  - Tensorflow, MatConvNet (Matlab), Keras (Wrapper for Tensorflow & Theano)

Image Classification



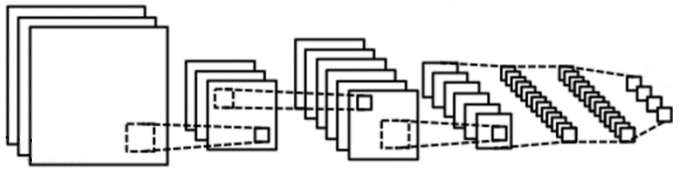
Common Implementation Using Caffe

Preparing Training Database



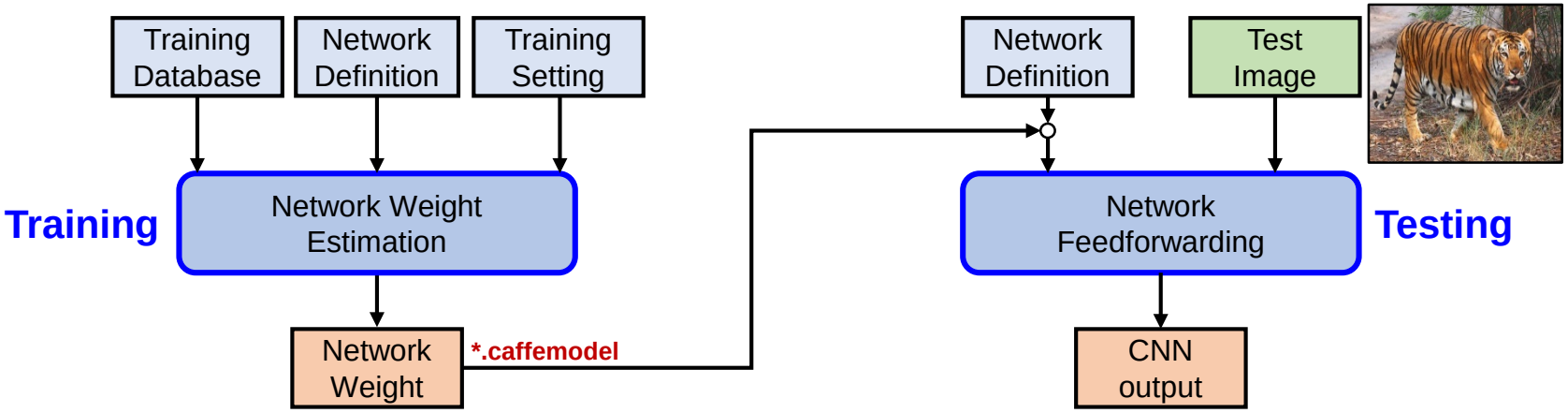
- **LMDB**
- LevelDB
- Raw images & labels

Defining a Network



- **Protobuf**
- Python

Protobuf: Protocol Buffers (Developed by Google)
LMDB: Lightning Memory-Mapped Database Manager



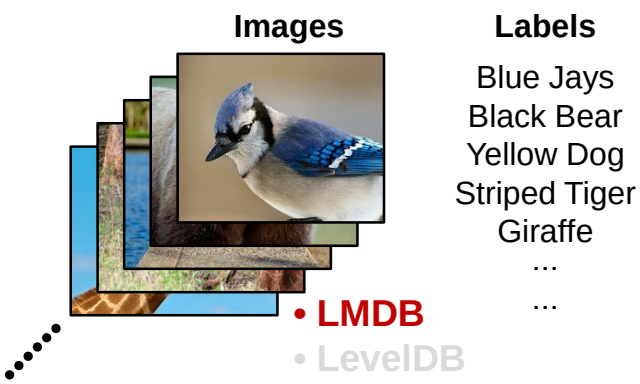
- **Caffe training executable + Protobuf**
  (only if network definition is done using protobuf)
- Python

- Caffe testing executable
  (only if network definition is done using protobuf)
- **Python**

Contents

Step 1: Preparing Training Database

Preparing Training Database



Defining a Network

• Protobuf
• Python

What is LMDB? (https://symas.com/lightning-memory-mapped-database/)
Lightning Memory-Mapped Database (LMDB) is a software library that provides a high-performance embedded transactional database in the form of a key-value store.

Training

Network Weight Estimation

Network Weight

*.caffemodel

• Caffe training executable + Protobuf
(only if network definition is done using protobuf)
• Python

Network Feedforwarding

CNN output

Testing

• Caffe testing executable
(only if network definition is done using protobuf)
• Python

Preparing Training Database

- Preparing annotation file

Task	Annotation	(Common) Annotation file
Image classification	Class of each image	txt file
Object detection (Bounding box estimation)	Position of each bounding box in an image	JSON file
Image segmentation (Pixel-wise labeling)	Class of each pixel in an image	Labeled image file

  - Better way for annotation?
    https://www.oreilly.com/ideas/data-preparation-in-the-age-of-deep-learning

- Converting annotated data(images & annotation file) into LMDB
  - Caffe tools: convert_imageset.cpp (for image classification example)
  - LMDB: a directory containing data.mdb, lock.mdb

JSON: JavaScript Object Notation
LMDB: Lightning Memory-Mapped Database Manager

Step 2: Defining a Network

Preparing Training Database

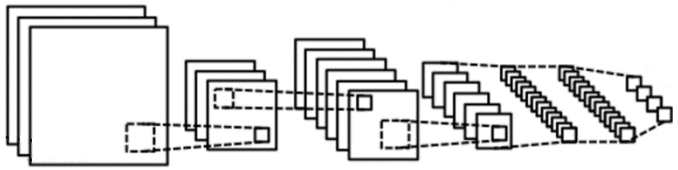
Images

Labels

Blue Jays
Black Bear
Yellow Dog
Striped Tiger
Giraffe
...
...

- LMDB
- LevelDB
- Raw images & labels

Defining a Network



- **Protobuf**
- Python

What is Protocol buffers? (https://developers.google.com/protocol-buffers/)
Protocol buffers are Google's language-neutral, platform-neutral, extensible mechanism for serializing structured data.

Training

Network Weight Estimation

Network Weight

*.cafffemodel

- Caffe training executable + Protobuf
  (only if network definition is done using protobuf)
- Python

Testing

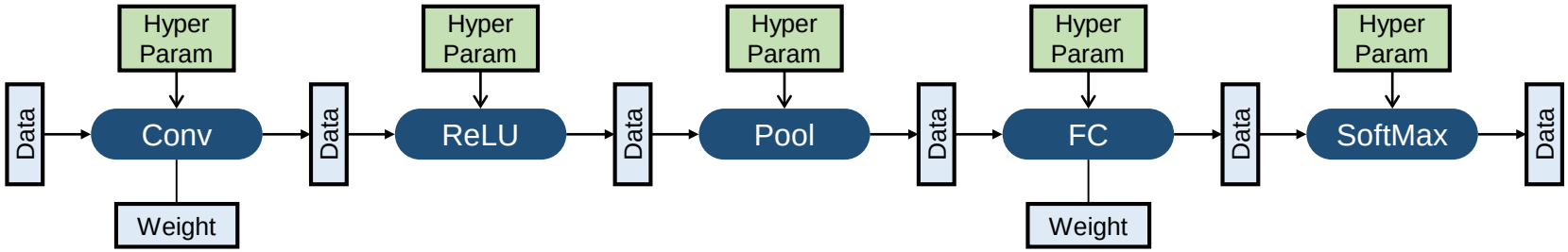
Network Feedforwarding

CNN output

- Caffe testing executable
  (only if network definition is done using protobuf)
- Python

CNN

- CNN: a Set of '**Data Transformation**'

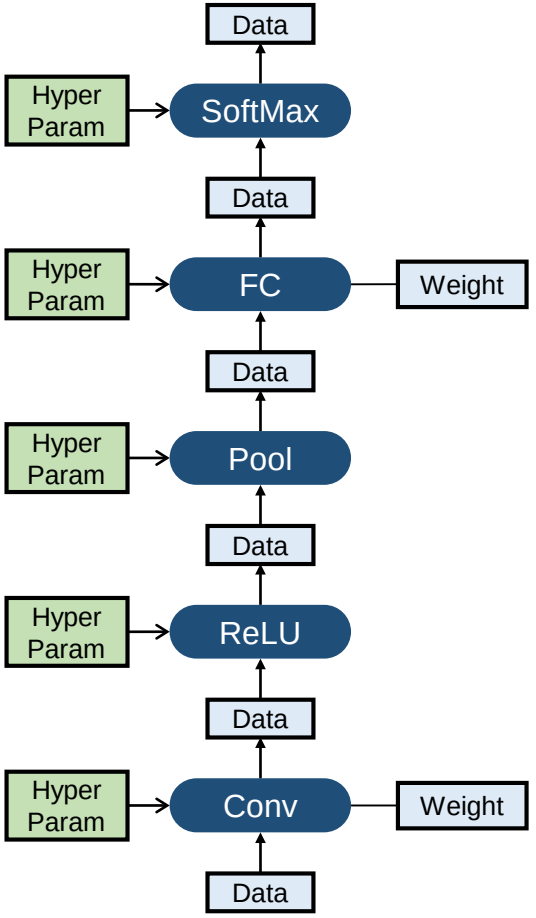


OOOO : Transformation

CNN

- CNN: a Set of '**Data Transformation**'

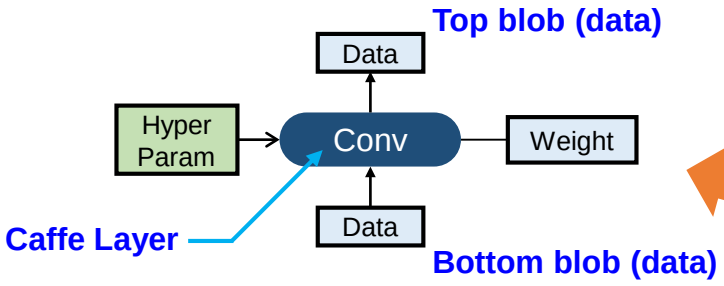
Just re-drawn as *bottom-top* style description for Caffe notation →

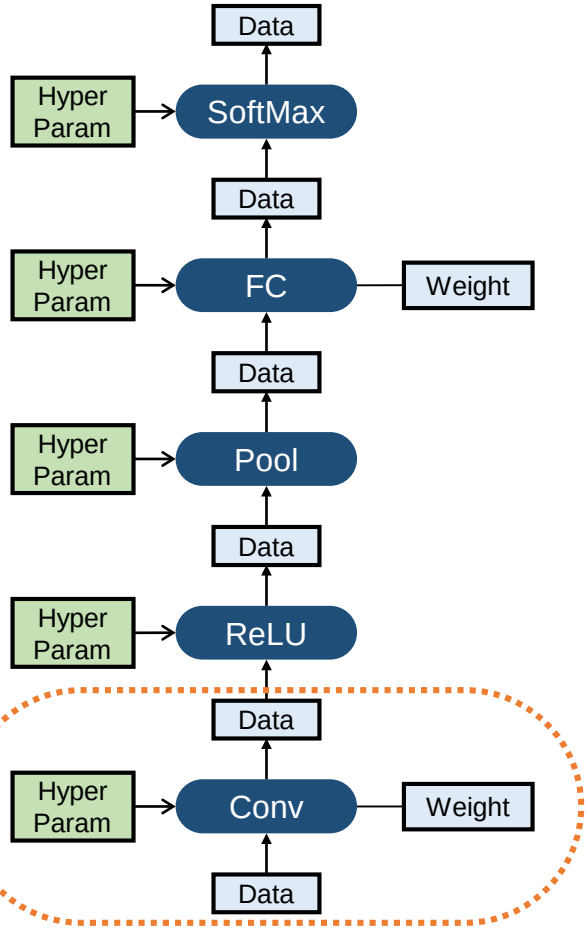


CNN in Caffe (1/5)

- CNN in Caffe: a Set of **Blob** for Data & **Caffe Layer** for Transformation
  - **Blob**
    - ✓ **N-D arrays** for storing values
    - ✓ Includes **data**, **weight** and their **diff** values
  - **Caffe Layer**
    - ✓ Fundamental computation unit in Caffe
    - ✓ Transforms bottom blobs to top blobs.
    - ✓ Not equal to conventional neural network layer.

    * Loading input data is also managed by Caffe layer.

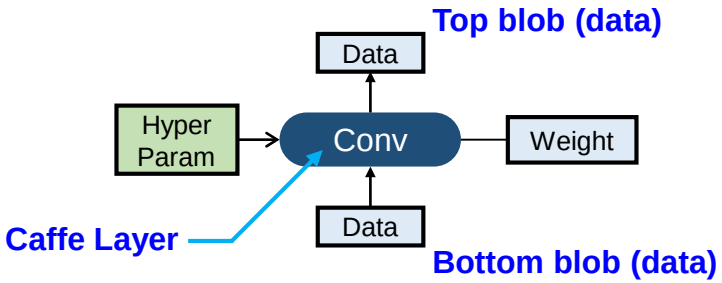




*Diff values are not drawn for brevity of the figure.

CNN in Caffe (2/5)

- Describing a **Caffe Layer** for Your CNN
  - Layer name (made by you)
  - Layer type
  - Bottom blob
  - Top blob
  - Hyper params



In your_cnn.prototxt,

```
...
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "data1"
  top: "conv1"
  convolution_param {
    num_output: 20
    kernel_size: 5
    weight_filler {
      type: "xavier"
    }
  }
}
...
```

• (Predefined) Caffe Layer Catalogue: http://caffe.berkeleyvision.org/tutorial/layers.html

CNN in Caffe (3/5) [optional]

- Where is a low-level code for a **Caffe Layer** ?
  - Layer (Each implemented by Class)
    - ✓ header:
      - include/caffe/layers/oooo_layer.hpp
    - ✓ source:
      - [For CPU] src/caffe/layers/oooo_layer.cpp
      - [For GPU] src/caffe/layers/oooo_layer.cu
    - ✓ major member functions
      - Reshape()
      - Forward() : Forward_cpu(), Forward_gpu()
      - Backward() : Backward_cpu(), Backward_gpu()
  - Hyper params
    - ✓ proto/caffe.proto

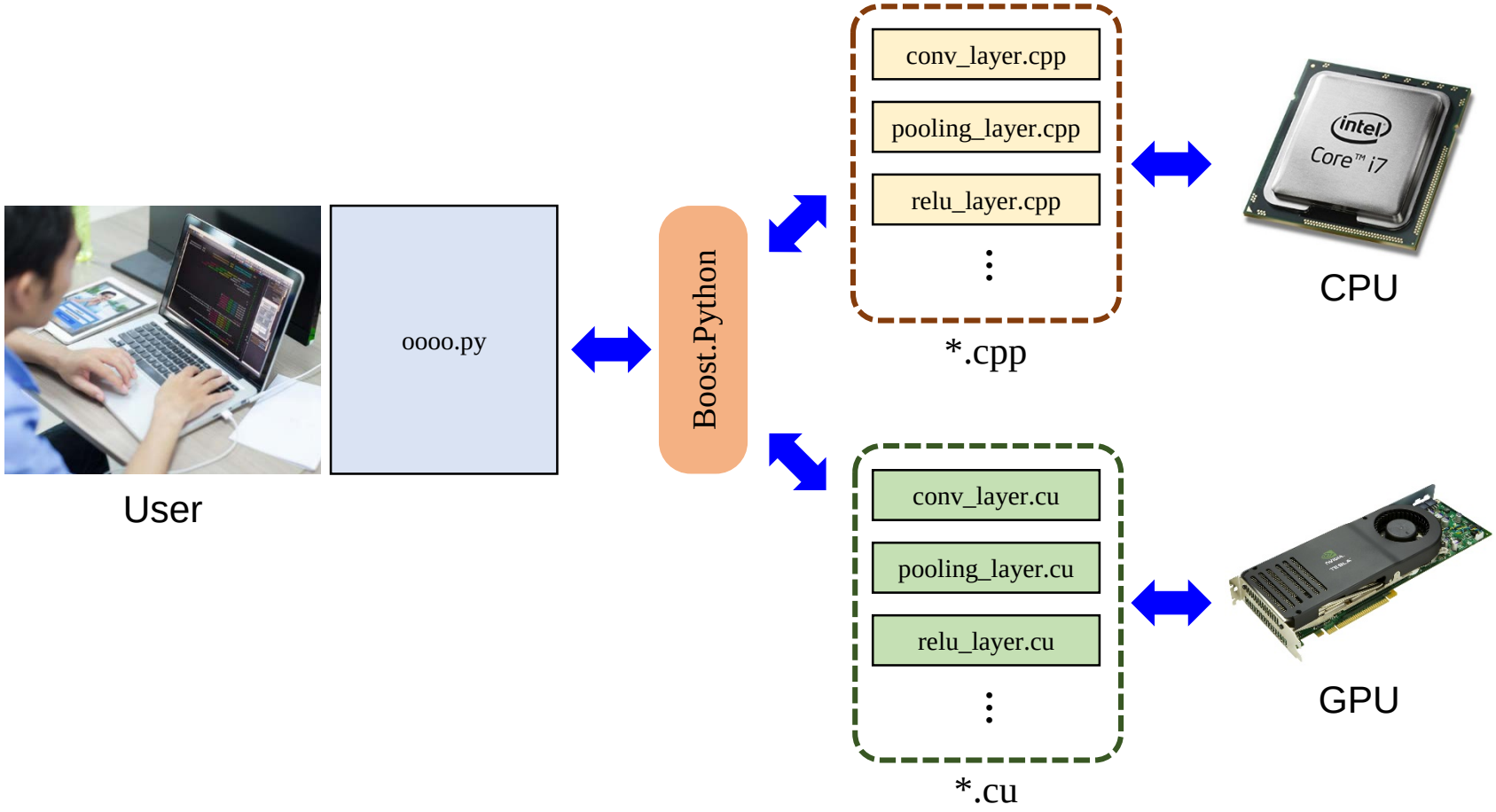
In your_cnn.prototxt,

```
...
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "data1"
  top: "conv1"
  convolution_param {
    num_output: 20
    kernel_size: 5
    weight_filler {
      type: "xavier"
    }
  }
}
...
```

- Developing new layers: https://github.com/BVLC/caffe/wiki/Development

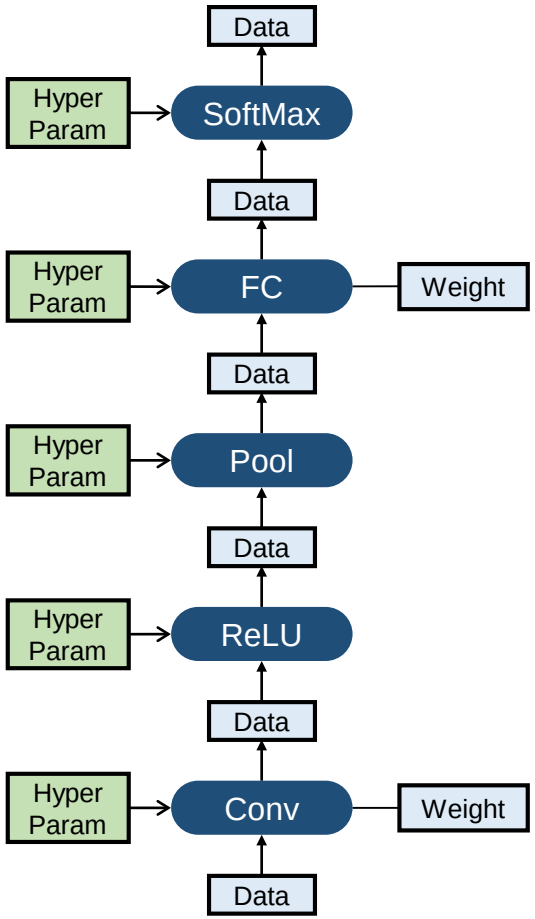
CNN in Caffe (4/5) [optional]

- How does python wrapper link with a **Caffe Layer** ?



CNN in Caffe (5/5)

- Defining a Complete CNN = Describing Whole **Caffe Layers** for the CNN



In your_cnn.prototxt,

```
layer {
  name: "data"
  type: "Data"
  top: "data"
  ...
}
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "data"
  top: "conv1"
  ...
}
layer {
  name: "relu1"
  type: "ReLU"
  bottom: "conv1"
  top: "conv1"
}
layer {
  name: "pool1"
  type: "Pooling"
  bottom: "conv1"
  top: "pool1"
  ...
}
layer {
  name: "fc1"
  type: "InnerProduct"
  bottom: "pool1"
  top: "fc1"
  ...
}
layer {
  name: "prob"
  type: "Softmax"
  bottom: "fc1"
  top: "prob"
}
```

Step 3: Training

Preparing Training Database

Images

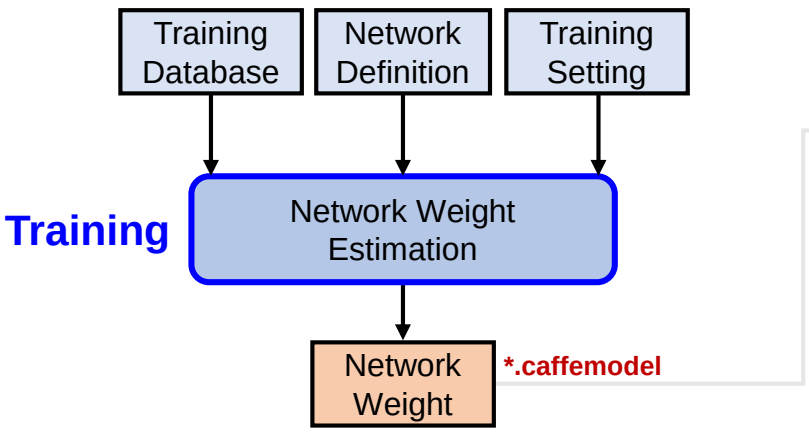
Labels

Blue Jays
Black Bear
Yellow Dog
Striped Tiger
Giraffe
...
...

• LMDB
• LevelDB
• Raw images & labels

Defining a Network

• Protobuf
• Python



• **Caffe training executable + Protobuf**
(only if network definition is done using protobuf)
• Python

• Caffe testing executable
(only if network definition is done using protobuf)
• Python

Training Using caffe Tool

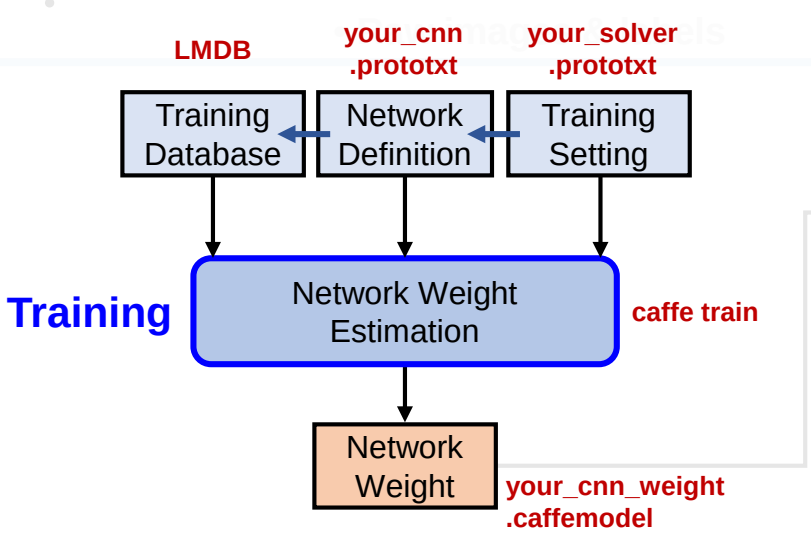
- One Line Command for Training
  - $ caffe train --solver=your_solver.prototxt

your_solver.prototxt

```
# The train/test net protocol buffer definition
train_net: "mnist/lenet_auto_train.prototxt"
test_net: "mnist/lenet_auto_test.prototxt"
# test_iter specifies how many forward passes the test should carry out.
test_iter: 100
# Carry out testing every 500 training iterations.
test_interval: 500
# The base learning rate, momentum and the weight decay of the network.
base_lr: 0.01
momentum: 0.9
weight_decay: 0.0005
# The learning rate policy
lr_policy: "inv"
gamma: 0.0001
power: 0.75
# Display every 100 iterations
display: 100
# The maximum number of iterations
max_iter: 10000
# snapshot intermediate results
snapshot: 5000
snapshot_prefix: “your_dir/lenet“
# solver mode: CPU or GPU
solver_mode: GPU
```

Training Using caffe Tool

```
$ caffe train --solver=your_solver.prototxt
```



Step 4: Testing

Preparing Training Database

Images

Labels

Blue Jays
Black Bear
Yellow Dog
Striped Tiger
Giraffe
...
...

• LMDB
• LevelDB
• Raw images & labels

Defining a Network

• Protobuf
• Python

Training

Training Database

Network Definition

Training Setting

Network Weight Estimation

Network Weight

*.caffemodel

• Caffe training executable + Protobuf
(only if network definition is done using protobuf)
• Python

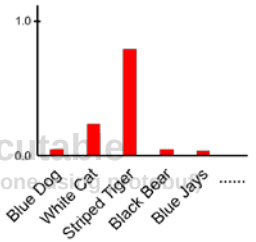
Testing

Network Definition

Test Image

Network Feedforwarding

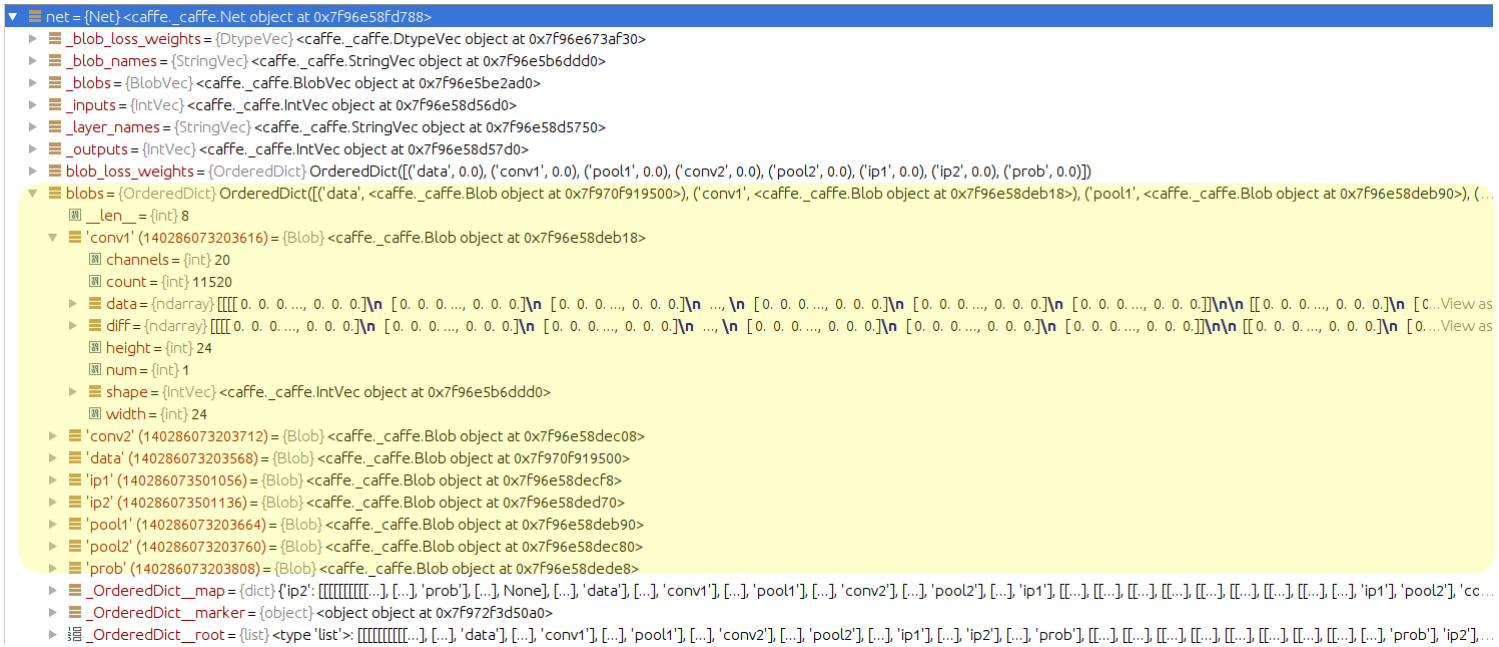
CNN output



• Caffe testing executable
(only if network definition is done using protobuf)
• Python

Your Network Loaded in Python

- Blobs : Stored in Python ‘ordered dictionary’
  - ex) net.blobs['conv1']
- Data in Blobs : Stored in Python NumPy’s ‘ndarray’
  - ex) net.blobs['conv1'].data



Monitoring a Network in PyCharm

Testing Using Python

- Procedure for Testing
  - (1) Load a network definition & its weight
  - (2) Load a test image & Put the image into the network
  - (3) Perform feedforwarding
  - (4) Get result

your_testing.py

```
import caffe
...
#(1)
model_def = caffe_root +'models/bvlc_alexnet/alexnet.prototxt'
model_weights = caffe_root + 'models/bvlc_alexnet/bvlc_alexnet.caffemodel'
net = caffe.Net(model_def, model_weights, caffe.TEST)
...
#(2)
image = caffe.io.load_image(caffe_root + 'examples/images/cat.jpg')
net.blobs['data'].data[...] = image
...
#(3)
output = net.forward()
...
#(4)
output_prob = output['prob'][0]
print 'predicted class is:', output_prob.argmax()
...
```

- Partially extracted from: http://nbviewer.jupyter.org/github/BVLC/caffe/blob/master/examples/00-classification.ipynb

Contents

What We Need for Caffe Programming

- Data Preparation
  - One of the most important things in deep learning
- Protobuf
  - Description grammar used for defining a network & descripting a training setting
- Python
  - Major language for most deep learning frameworks including Caffe

References (1/3)

- General Introduction
  - Stanford cs231n
    http://vision.stanford.edu/teaching/cs231n/slides/2015/caffe_tutorial.pdf
  - DIY Deep Learning for Vision-a Hands-On Tutorial with Caffe
    http://tutorial.caffe.berkeleyvision.org/
  - A Practical Introduction to Deep Learning with Caffe
    http://www.panderson.me/images/Caffe.pdf

- PyCaffe
  - Deep learning tutorial on Caffe technology : basic commands, Python and C++ code
    http://christopher5106.github.io/deep/learning/2015/09/04/Deep-learning-tutorial-on-Caffe-Technology.html
  - Deep Learning With Caffe In Python
    https://prateekvjoshi.com/2016/02/02/deep-learning-with-caffe-in-python-part-i-defining-a-layer/

References (2/3)

- PyCaffe (Conti.)
  - Tutorial for pycaffe, the Python API to the Neural Network framework, Caffe
    https://github.com/nitnelave/pycaffe_tutorial
  - A Practical Introduction to Deep Learning with Caffe and Python
    http://adilmoujahid.com/posts/2016/06/introduction-deep-learning-python-caffe/
  - pyCaffe Tools, Examples and Resources
    http://davidstutz.de/pycaffe-tools-examples-and-resources/
    https://github.com/davidstutz/caffe-tools
  - Training Multi-Layer Neural Network with Caffe
    http://nbviewer.jupyter.org/github/joyofdata/joyofdata-articles/blob/master/deeplearning-with-caffe/Neural-Networks-with-Caffe-on-the-GPU.ipynb

References (3/3)

- Caffe Layer
  - Developing new layers
    https://github.com/BVLC/caffe/wiki/Development
  - Making a Caffe Layer
    https://chrischoy.github.io/research/making-caffe-layer/
  - How to create your own layer in deep learning framework CAFFE
    https://yunmingzhang.wordpress.com/2015/01/19/how-to-create-your-own-layer-in-deep-learning-framework-caffe/
  - Simple Example: Sin Layer
    https://github.com/BVLC/caffe/wiki/Simple-Example:-Sin-Layer