INTEL OPTIMIZED CAFFE

Agenda

1. Activity 0: Installing Intel-optimized and non-optimized Caffe
2. Activity 1: optimized Caffe training
3. Model parameters change
4. Activity 2: scaling with batch size
5. Activity 3: non-optimized Caffe training
6. Activity 4: non-optimized Caffe scaling with batch size

Activity 0:

Intel Optimized Caffe installation:

```
conda create -y -n caffe_intel -c intel python=3 caffe
```

Non optimized Caffe installation:

```
conda create -y -n caffe_simple python=3 caffe
```

Caffe tutorial

https://software.intel.com/en-us/articles/training-and-deploying-deep-learning-networks-with-caffe-optimized-for-intel-architecture

- Summary
- Installation
- Data layer
- Dataset preparation
- Training
- Multinode distributed training
- Fine-tuning
- Testing
- Feature extractor and visualization
- Using the Python* API
- Debugging
- Examples
- Current Caffe* usages
- Further reading

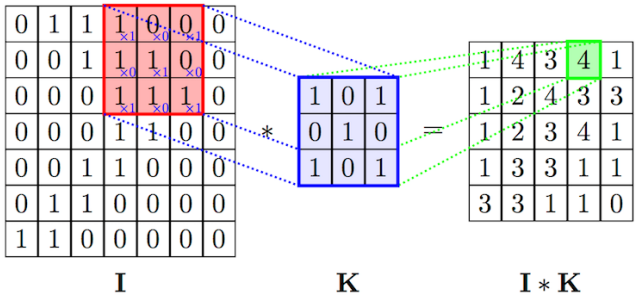
MNIST

Train images: 60000

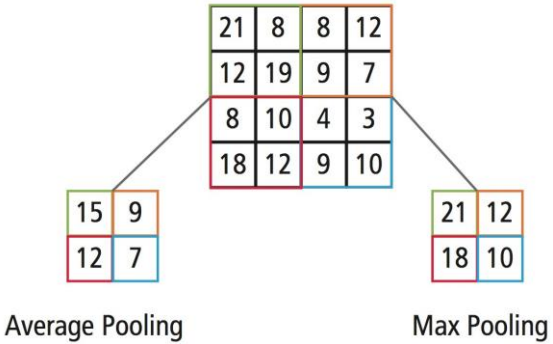
Test images: 10000

Layers:

Convolutional

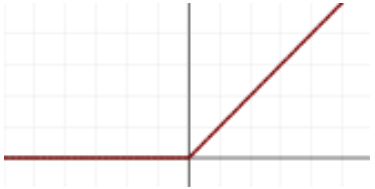


Pooling

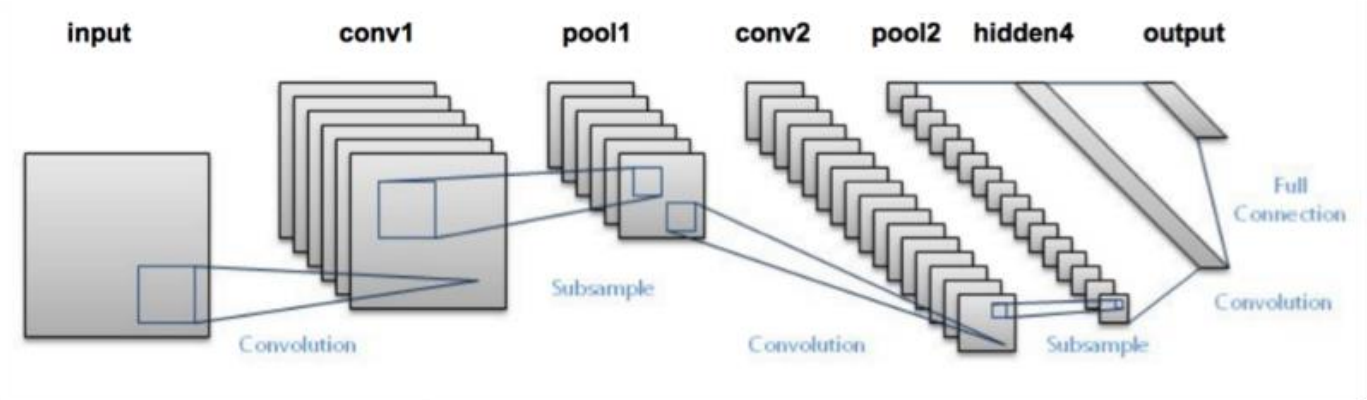


ReLU

$$a = \max(z, 0)$$



LeNet model



Caffe models structures

model.prototxt

```
layer {
        name: “layer_name”
        type: “”
        bottom: “name_of_input_layer”
        top: “name_of_output_layer”
        layer_params{
        …
        }
}
…..
```

solver.prototxt

```
net: "model.prototxt"
base_lr: 0.01
lr_policy: "step"
stepsize: 320000
gamma: 0.96
max_iter: 10000
momentum: 0.9
weight_decay: 0.0002
```

Model parameters:

- Network architecture
- Batch size
- Optimizer: SGD, Adam, Adadelta…
- Learning rate

Activity 1: environment setting

1. Go to work folder:

```
cd mnist/
```

2. Create folder for saving models:

```
mkdir snapshots
```

Parameters changing:

1. For batch size changing: open **`lenet_train_test.prototxt`** file and change batch_size parameter in the input layer (img. 1).
2. For ateps count changing: open file **`lenet_solver.prototxt`** and change max_iter parameter(img. 2).

```
name: "LeNet"
layer {
  name: "mnist"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TRAIN
  }
  transform_param {
    scale: 0.00390625
  }
  data_param {
    source: "mnist_train_lmdb"
    batch_size: 128
    backend: LMDB
  }
```

Img.1

```
# The train/test net protocol buffer definition
net: "lenet_train_test.prototxt"
# test_iter specifies how many forward passes the test should carry out.
# In the case of MNIST, we have test batch size 100 and 100 test iterations,
# covering the full 10,000 testing images.
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: 0.001
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: 5000
# snapshot intermediate results
snapshot: 5000
snapshot_prefix: "snapshots/lenet"
# solver mode: CPU or GPU
solver_mode: CPU
```

Img. 2

Activity 2: non-optimized Caffe training

1. Activate virtual environment with non-optimized Caffe:

   ```
   conda activate caffe_simple
   ```

2. Change `batch_size=16, max_iter=40000` and launch training:

   ```
   time caffe train --solver=lenet_solver.prototxt
   ```

3. Change `batch_size=64, max_iter=10000` and launch training:

   ```
   time caffe train --solver=lenet_solver.prototxt
   ```

4. Change `batch_size=256, max_iter=2500` and launch training:

   ```
   time caffe train --solver=lenet_solver.prototxt
   ```

5. `conda deactivate`

You can launch training with only one parameters combination.

Activity 3: optimized Caffe scaling

1. Activate virtual environment with optimized Caffe:

   ```
   conda activate caffe_intel
   ```

2. Change `batch_size=16, max_iter=40000` and launch training:

   ```
   time caffe train --solver=lenet_solver.prototxt
   ```

3. Change `batch_size=64, max_iter=10000` and launch training:

   ```
   time caffe train --solver=lenet_solver.prototxt
   ```

4. Change `batch_size=256, max_iter=2500` and launch training:

   ```
   time caffe train --solver=lenet_solver.prototxt
   ```

5. `conda deactivate`

Results:

	batch_size = 16	batch_size = 64	batch_size = 256
Caffe	11m 30s	7m 53s	9m 55s
Intel Caffe	2m 58s	2m 0s	1m 46s

Optimized Caffe advantages:

- Better scaling
- Faster training

Resources

Intel Optimized Caffe github:

https://github.com/intel/caffe

Caffe tutorial:

https://software.intel.com/en-us/articles/training-and-deploying-deep-learning-networks-with-caffe-optimized-for-intel-architecture

Multinode training

https://github.com/intel/caffe/wiki/Multinode-guide


intel
Software