NVIDIA®

GETTING STARTED WITH DIGITS™

Allison Gray

Solutions Architect

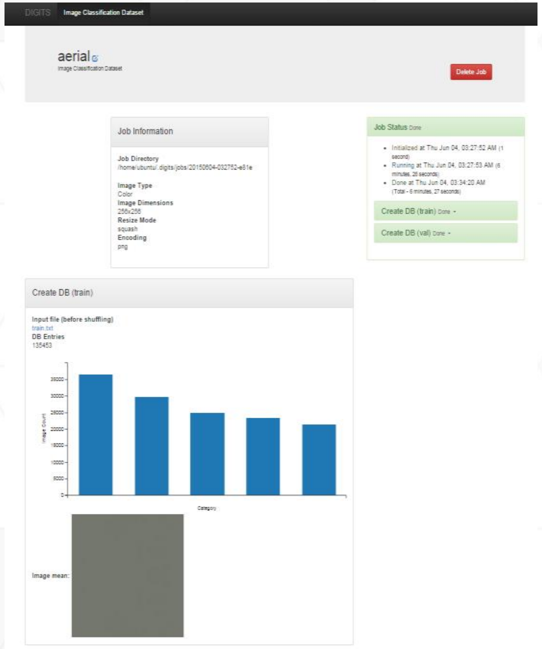
AGENDA

- INTRODUCTION TO DIGITS AND IMAGE CLASSIFICATION
- HOW TO INSTALL DIGITS
- CREATING DATASETS
- TRAIN A NETWORK
- MODIFY YOUR NETWORK

NVIDIA DIGITS

Interactive Deep Learning GPU Training System

Process Data



DIGITS | Image Classification Dataset

aerial
Image Classification Dataset

Delete Job

Job Information

Job Directory
/home/ubuntu/.digits/jobs/20150604-032752-e81e

Image Type
Color
Image Dimensions
256x256
Resize Mode
squash
Encoding
png

Job Status Done

- Initialized at Thu Jun 04, 03:27:52 AM (1 second)
- Running at Thu Jun 04, 03:27:53 AM (6 minutes, 26 seconds)
- Done at Thu Jun 04, 03:34:20 AM (Total - 6 minutes, 27 seconds)

Create DB (train) Done

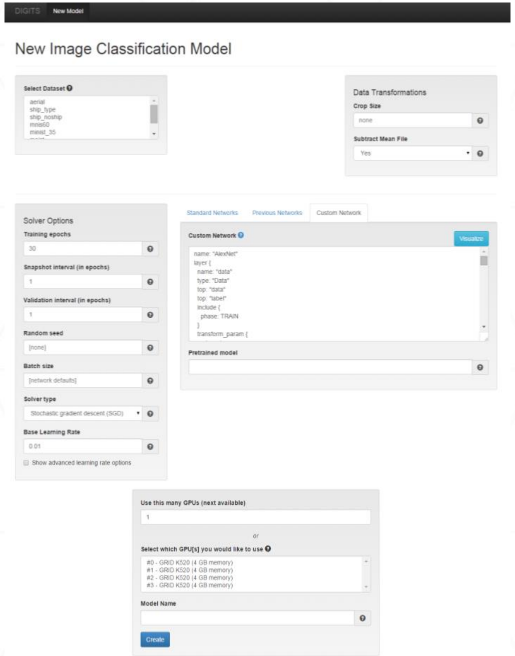
Create DB (val) Done

Create DB (train)

Input file (before shuffling)
train.txt
DB Entries
135453

Image mean:

Configure DNN



DIGITS | New Model

New Image Classification Model

Select Dataset
aerial
ship_type
ship_noship
mnist60
minist_35

Data Transformations
Crop Size: none
Subtract Mean File: Yes

Solver Options
Training epochs: 30
Snapshot interval (in epochs): 1
Validation interval (in epochs): 1
Random seed: [none]
Batch size: [network defaults]
Solver type: Stochastic gradient descent (SGD)
Base Learning Rate: 0.01
□ Show advanced learning rate options

Standard Networks | Previous Networks | Custom Network

Custom Network

Visualize

```
name: "AlexNet"
layer {
  name: "data"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TRAIN
  }
  transform_param {
```

Pretrained model

Use this many GPUs (next available): 1

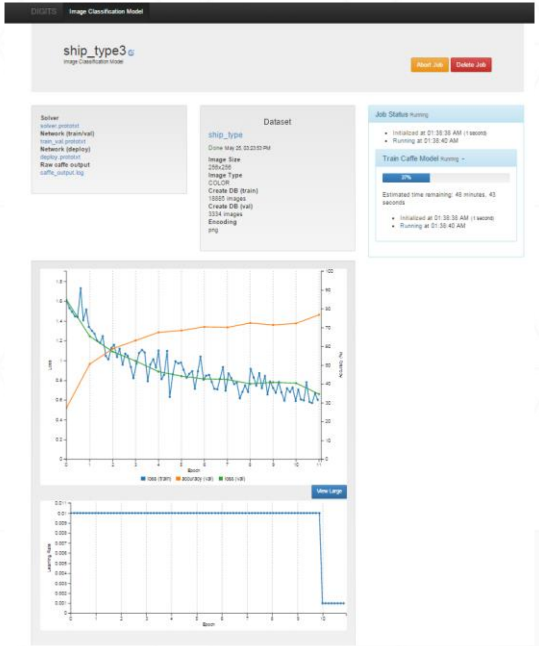
or

Select which GPU[s] you would like to use
#0 - GRID K520 (4 GB memory)
#1 - GRID K520 (4 GB memory)
#2 - GRID K520 (4 GB memory)
#3 - GRID K520 (4 GB memory)

Model Name

Create

Monitor Progress



DIGITS | Image Classification Model

ship_type3
Image Classification Model

Abort Job | Delete Job

Solver
solver.prototxt
Network (train/val)
train_val.prototxt
Network (deploy)
deploy.prototxt
Raw caffe output
caffe_output.log

Dataset
ship_type
Done May 25, 03:23:53 PM
Image Size
256x256
Image Type
COLOR
Create DB (train)
18885 images
Create DB (val)
3334 images
Encoding
png

Job Status Running

- Initialized at 01:38:38 AM (1 second)
- Running at 01:38:40 AM

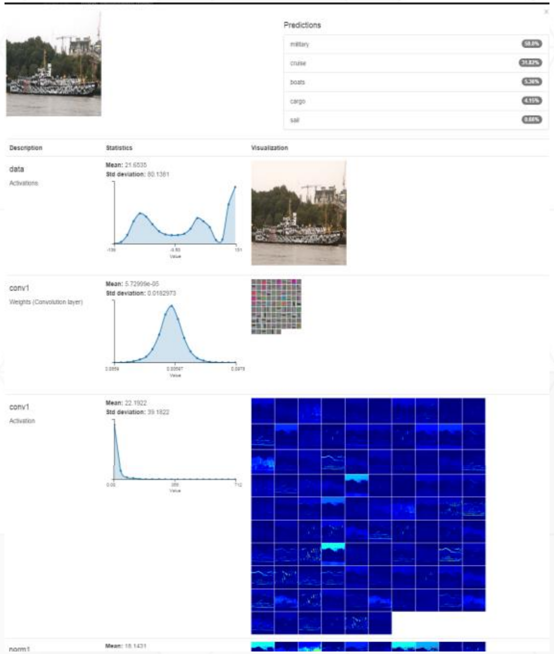
Train Caffe Model Running

37%

Estimated time remaining: 48 minutes, 43 seconds

- Initialized at 01:38:38 AM (1 second)
- Running at 01:38:40 AM

Visualization



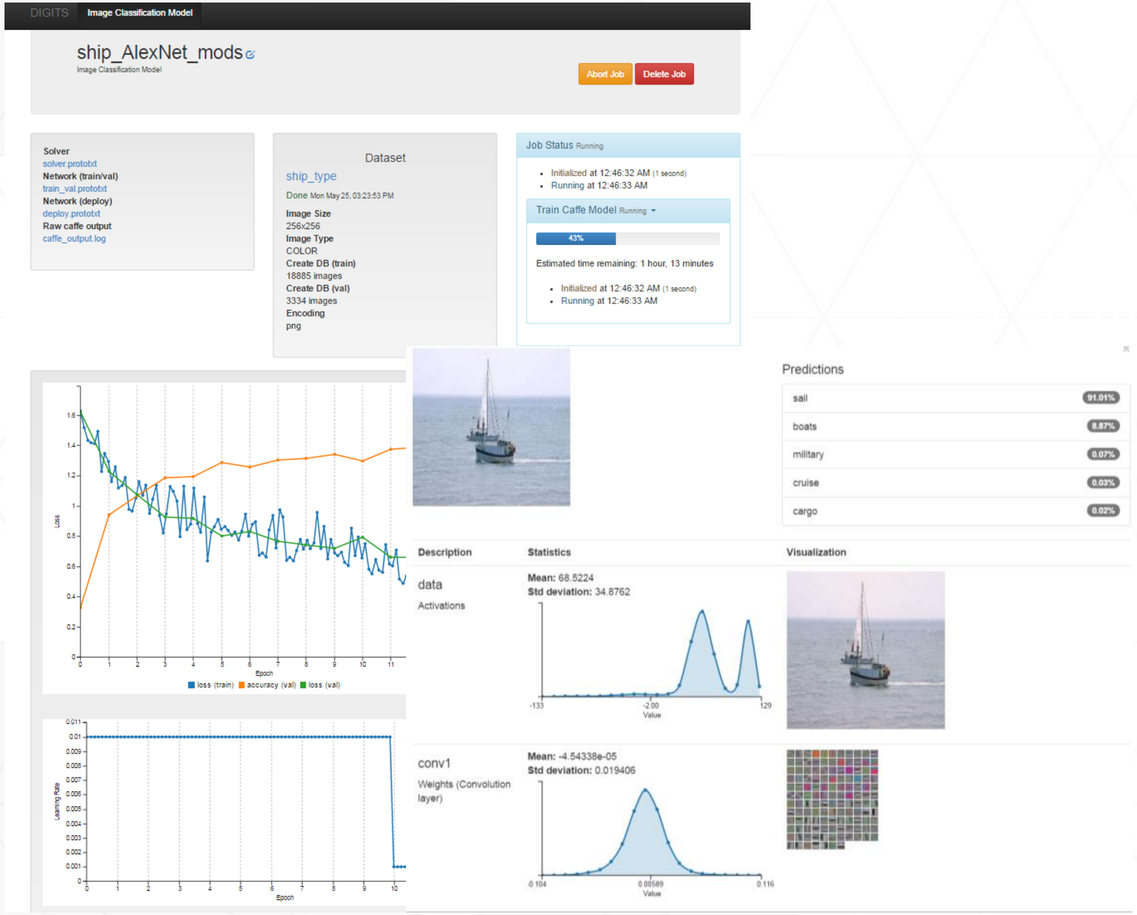
Predictions

military	58.0%
cruise	31.82%
boats	5.36%
cargo	4.15%
sail	0.66%

Description	Statistics	Visualization
data Activations	Mean: 21.6535 Std deviation: 80.1381	
conv1 Weights (Convolution layer)	Mean: 5.72999e-05 Std deviation: 0.0182973	
conv1 Activation	Mean: 22.1922 Std deviation: 39.1822	
norm1	Mean: 18.1431	

NVIDIA DIGITS

Who is DIGITS for?


DIGITS
Image Classification Model
Abort Job
Delete Job
Epoch
Loss
■ loss (train) ■ accuracy (val) ■ loss (val)
Learning Rate
Epoch

ship_AlexNet_mods
Image Classification Model

Solver
solver.prototxt
Network (train/val)
train_val.prototxt
Network (deploy)
deploy.prototxt
Raw caffe output
caffe_output.log

Dataset
ship_type
Done Mon May 25, 03:23:53 PM
Image Size
256x256
Image Type
COLOR
Create DB (train)
18885 images
Create DB (val)
3334 images
Encoding
png

Job Status Running

- Initialized at 12:46:32 AM (1 second)
- Running at 12:46:33 AM

Train Caffe Model Running

43%

Estimated time remaining: 1 hour, 13 minutes

- Initialized at 12:46:32 AM (1 second)
- Running at 12:46:33 AM

Predictions

sail	91.01%
boats	8.87%
military	0.07%
cruise	0.03%
cargo	0.02%

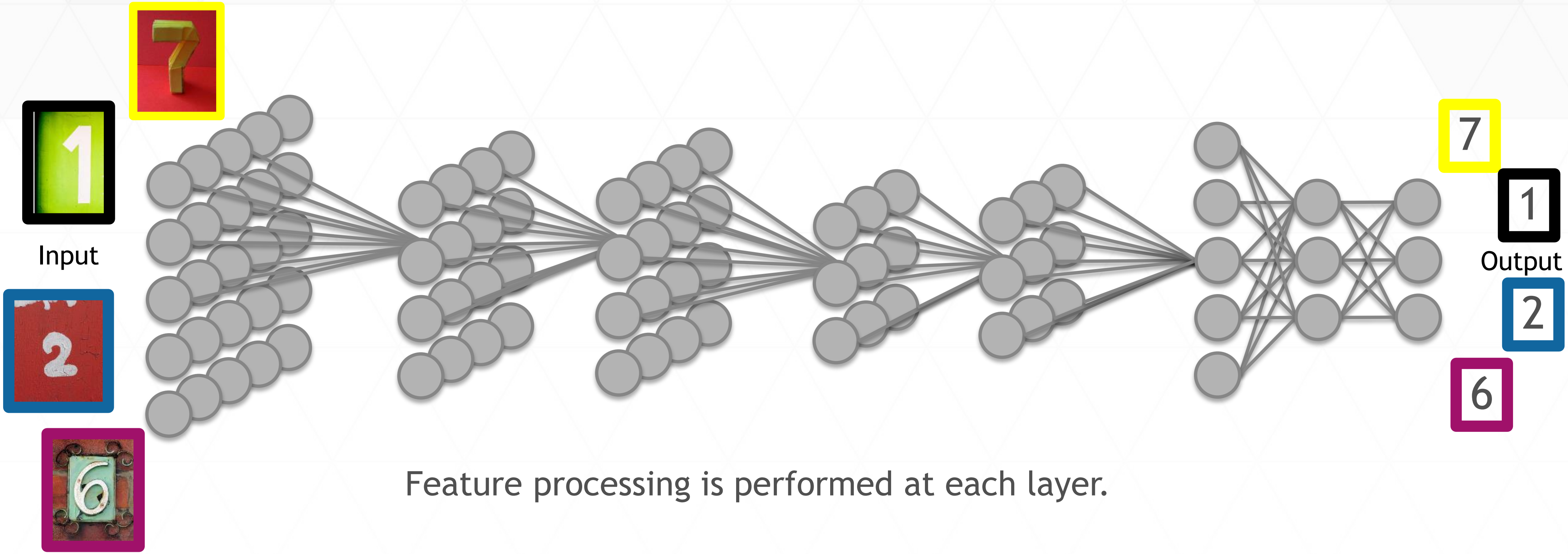
Description	Statistics	Visualization
data Activations	Mean: 68.5224 Std deviation: 34.8762 -133 -2.00 129 Value	
conv1 Weights (Convolution layer)	Mean: -4.54338e-05 Std deviation: 0.019406 -0.104 0.00589 0.116 Value	

Data Scientists & Researchers:

- Quickly design the best deep neural network (DNN) for your data
- Monitor DNN training quality in real-time
- Manage training of many DNNs in parallel on multi-GPU systems, and multi-GPU training

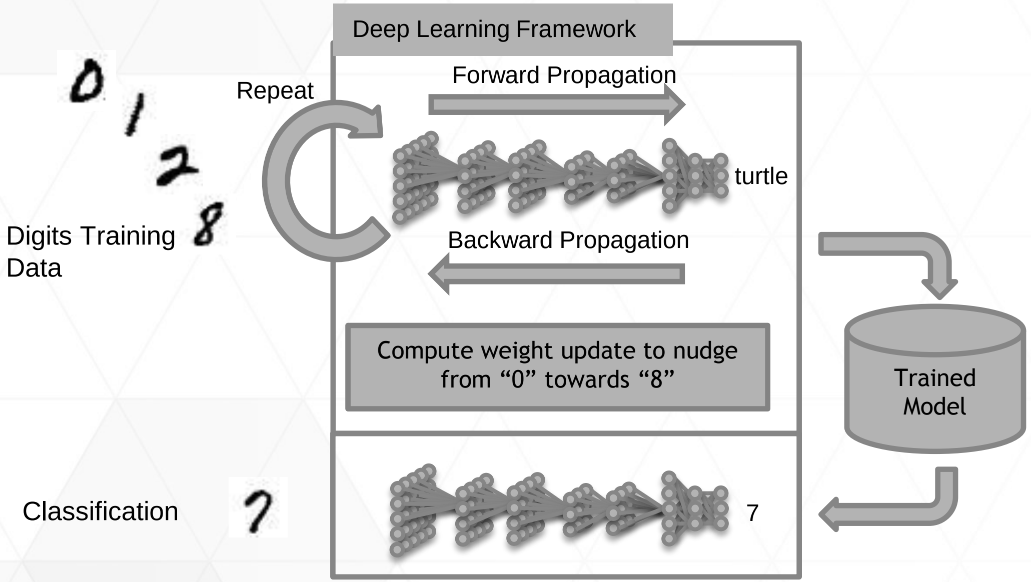
WHAT IS DEEP LEARNING?

A Classification Example



Feature processing is performed at each layer.

IMAGE CLASSIFICATION WITH DNNS



- Example training process
  - Pick a DNN design
  - Input training images
- Test accuracy
  - If bad: modify DNN, fix training set or update training parameters

HOW DO I GET DIGITS?

Two ways to install it

- Easy way
  - OS - Ubuntu 14.04
  - Download the web installer - https://developer.nvidia.com/digits
- Not as easy way
  - OSX, Windows(untested)
  - Build NVIDIA Caffe branch - https://github.com/NVIDIA/caffe
  - Download DIGITS from github – https://github.com/NVIDIA/DIGITS
- Hardware/software recommendations
  - GPU(s) with compute capabilities >=3.0 for cuDNN support
  - OS - Ubuntu 14.04

NVIDIA DIGITS

Hands-on Lab

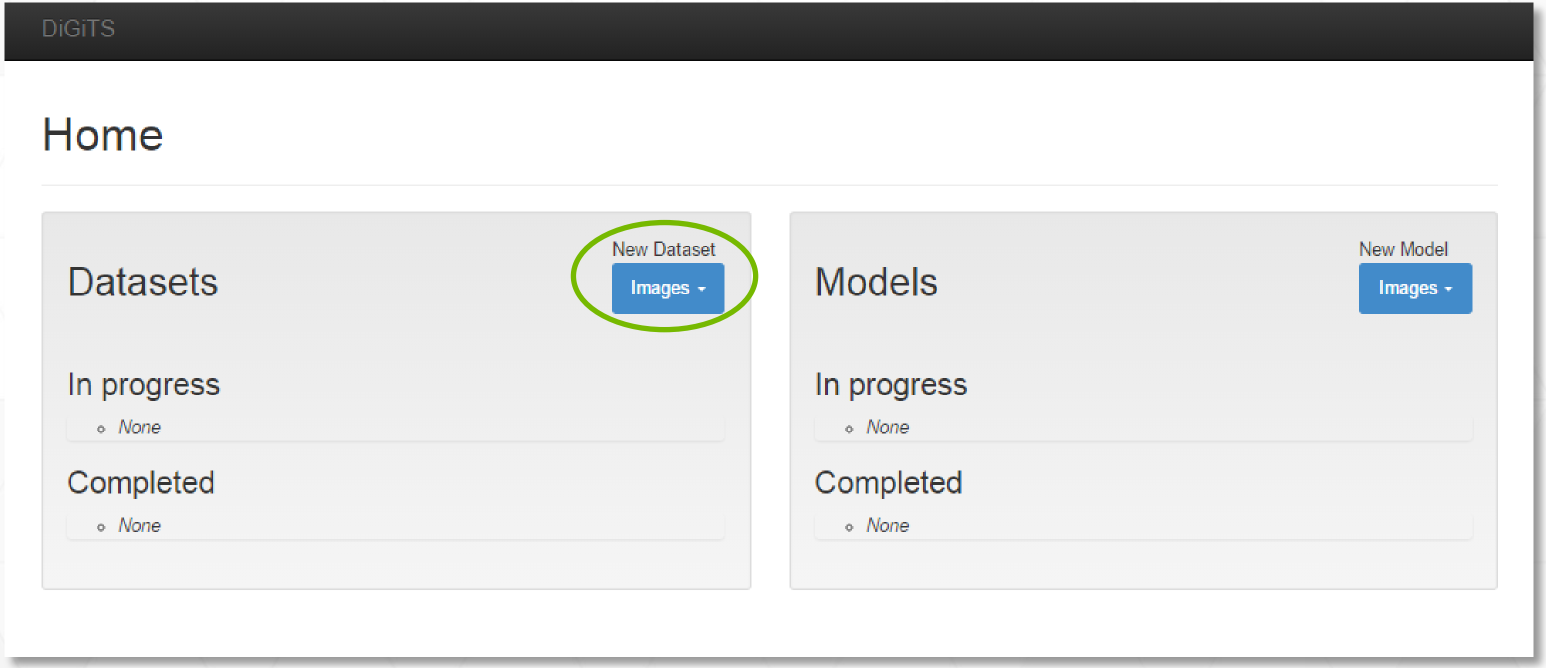
Quickly train and improve a DNN well suited for your data

- Create a database
- Start out with a standard network and start training
  - Classify images to further understand performance
- Modify your network
- Adding/removing parameters
- Inflating your data

NVIDIA DIGITS

Creating your Dataset

Main Console



DiGiTS

Home

Datasets

New Dataset

Images ▾

In progress

- *None*

Completed

- *None*

Models

New Model

Images ▾

In progress

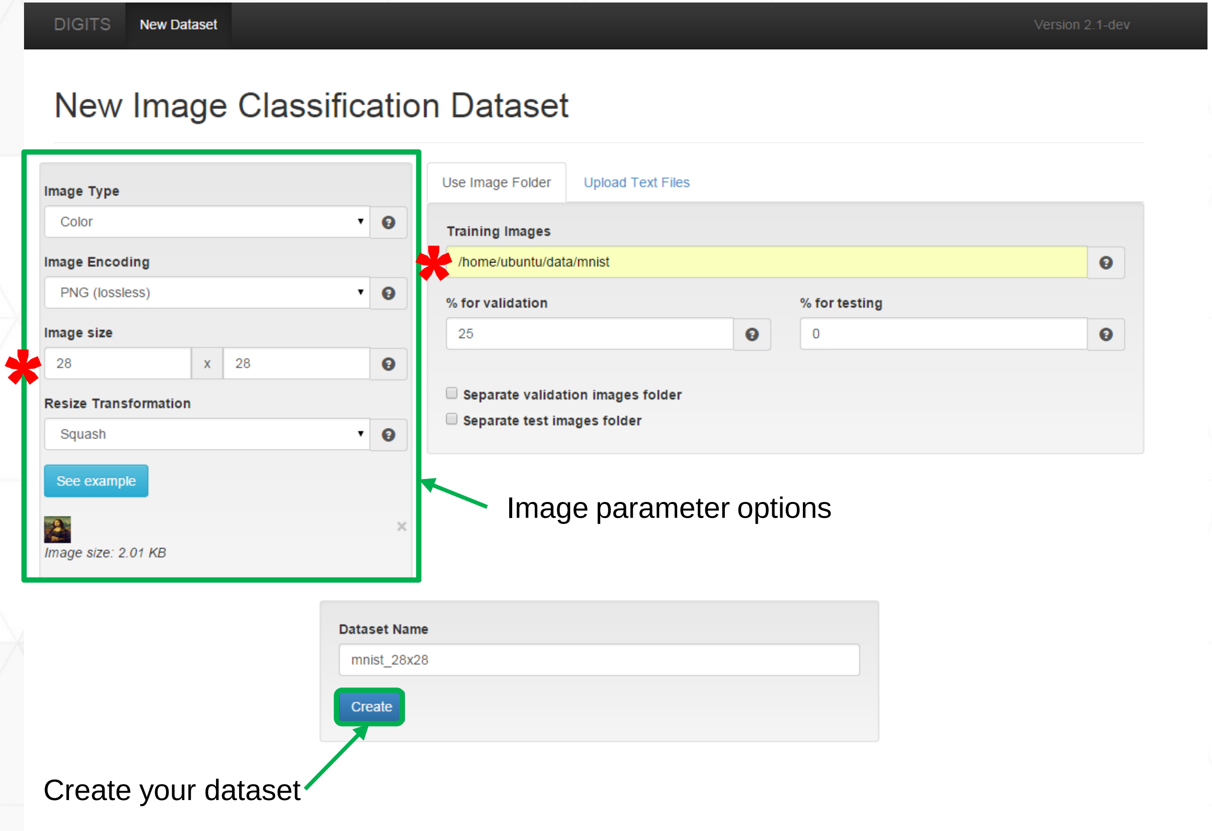
- *None*

Completed

- *None*

NVIDIA DIGITS

Create Database



New Image Classification Dataset

Image Type: Color

Image Encoding: PNG (lossless)

Image size: 28 x 28

Resize Transformation: Squash

Image size: 2.01 KB

Training Images: /home/ubuntu/data/mnist

% for validation: 25

% for testing: 0

□ Separate validation images folder

□ Separate test images folder

Dataset Name: mnist_28x28

Image parameter options

Create your dataset

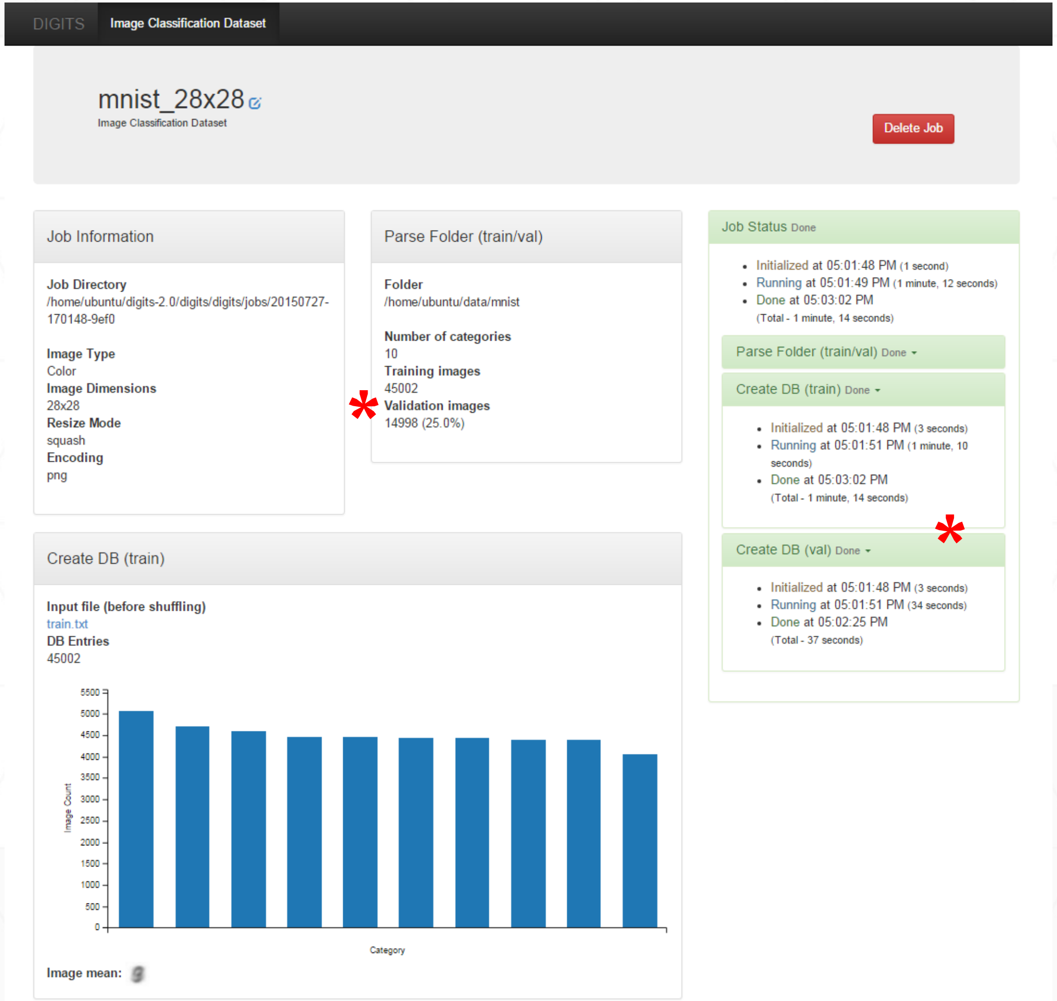
Input Data Format

```
/path/to/images/
              |-0/
              |      |-0_0jpg
              |      |-0_1.jpg
              |-1/
              |      |1_0.jpg
              |      |1_1.jpg
              |-2/
                     |-2_0.jpg
                     |-2_1.jpg
```

* Lab tasks

NVIDIA DIGITS

Database results



mnist_28x28

Image Classification Dataset

Job Information

Job Directory
/home/ubuntu/digits-2.0/digits/digits/jobs/20150727-170148-9ef0

Image Type
Color

Image Dimensions
28x28

Resize Mode
squash

Encoding
png

Parse Folder (train/val)

Folder
/home/ubuntu/data/mnist

Number of categories
10

Training images
45002

* **Validation images**
14998 (25.0%)

Create DB (train)

Input file (before shuffling)
train.txt

DB Entries
45002

Image mean:

Job Status Done

- Initialized at 05:01:48 PM (1 second)
- Running at 05:01:49 PM (1 minute, 12 seconds)
- Done at 05:03:02 PM
  (Total - 1 minute, 14 seconds)

Parse Folder (train/val) Done

Create DB (train) Done

- Initialized at 05:01:48 PM (3 seconds)
- Running at 05:01:51 PM (1 minute, 10 seconds)
- Done at 05:03:02 PM
  (Total - 1 minute, 14 seconds)

*

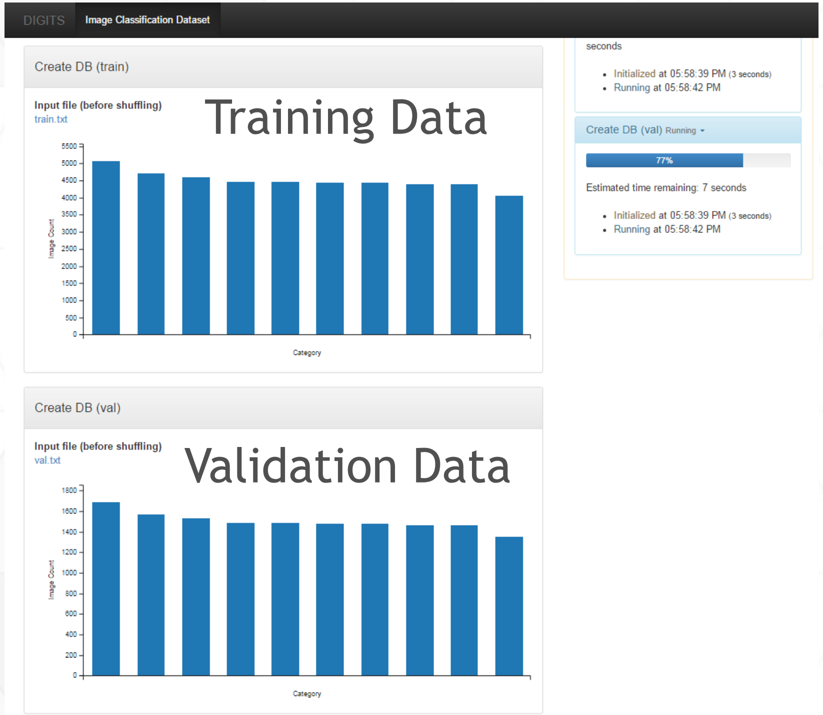
Create DB (val) Done

- Initialized at 05:01:48 PM (3 seconds)
- Running at 05:01:51 PM (34 seconds)
- Done at 05:02:25 PM
  (Total - 37 seconds)

* Lab tasks

TRAINING AND VALIDATION DATA

Why do we need it?



Create DB (train)

Input file (before shuffling)
train.txt

Training Data

seconds

- Initialized at 05:58:39 PM (3 seconds)
- Running at 05:58:42 PM

Create DB (val) Running

77%

Estimated time remaining: 7 seconds

- Initialized at 05:58:39 PM (3 seconds)
- Running at 05:58:42 PM

Create DB (val)

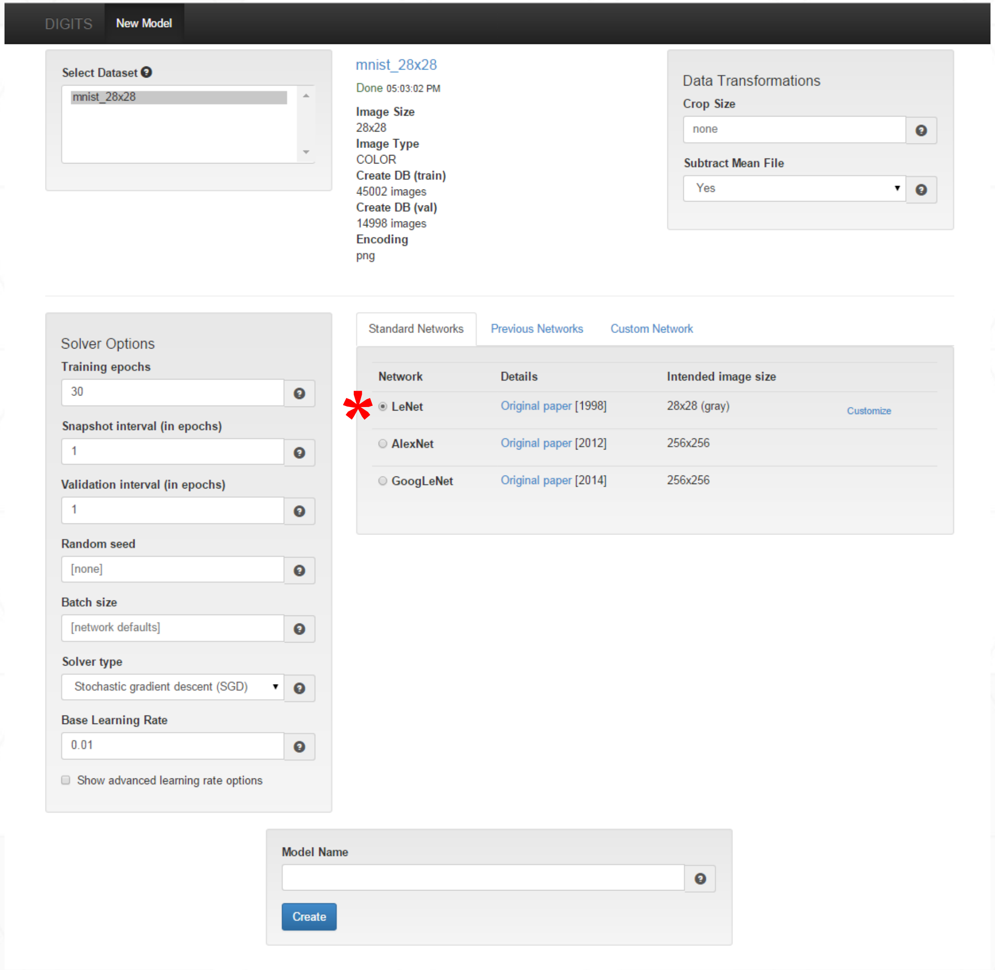
Input file (before shuffling)
val.txt

Validation Data

- Training data is used to train our neural network
  - Teaches the network to classify object categories
- Validation data tests the performance of the network
  - This data is only used for testing the generalization ability of the network
  - Not used to teach/train network
  - Prevents use of and identifies when network is overfitting

NVIDIA DIGITS

Network Configuration – Start with a Standard Network



Select Dataset
mnist_28x28

mnist_28x28
Done 05:03:02 PM
Image Size
28x28
Image Type
COLOR
Create DB (train)
45002 images
Create DB (val)
14998 images
Encoding
png

Data Transformations
Crop Size
none
Subtract Mean File
Yes

Solver Options
Training epochs
30
Snapshot interval (in epochs)
1
Validation interval (in epochs)
1
Random seed
[none]
Batch size
[network defaults]
Solver type
Stochastic gradient descent (SGD)
Base Learning Rate
0.01
☐ Show advanced learning rate options

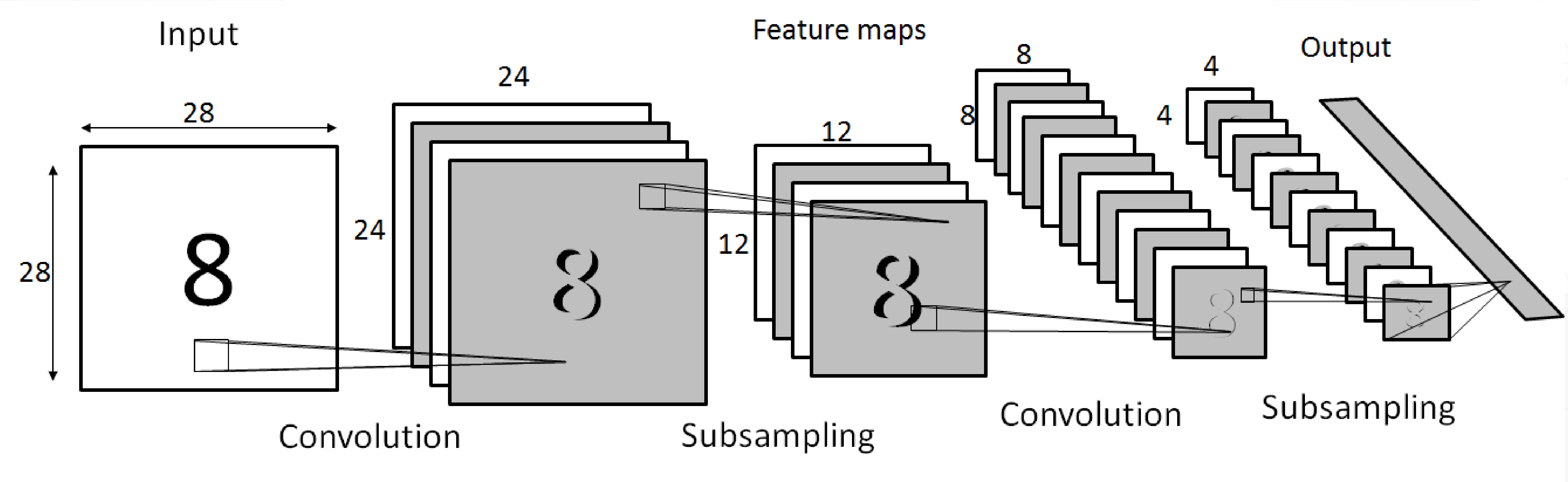
Network	Details	Intended image size	
* ● LeNet	Original paper [1998]	28x28 (gray)	Customize
○ AlexNet	Original paper [2012]	256x256	
○ GoogLeNet	Original paper [2014]	256x256	

Model Name

* Lab tasks

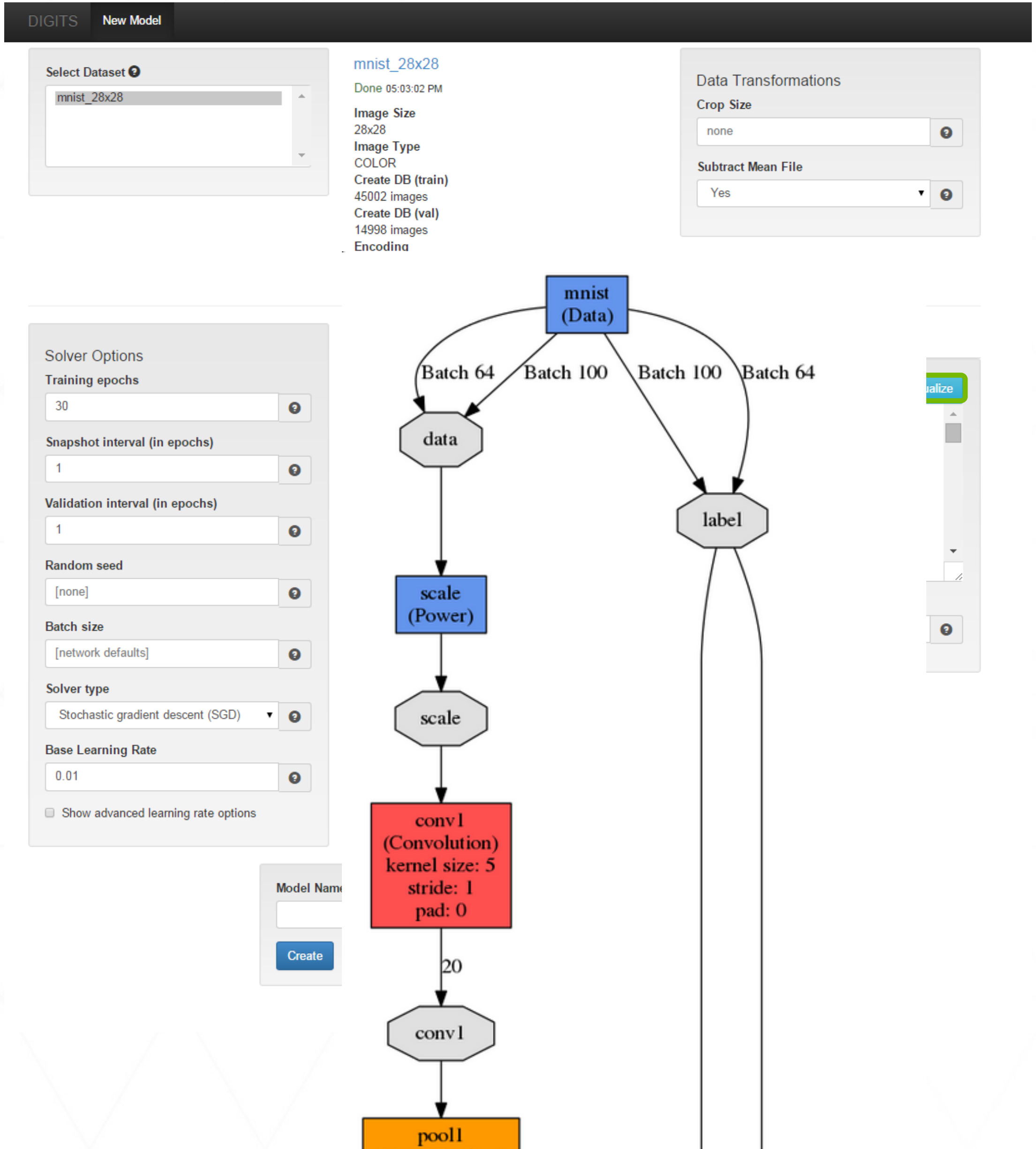
LENET

Network Configuration



NVIDIA DIGITS

Visualizing your Network



DIGITS | New Model

Select Dataset
mnist_28x28

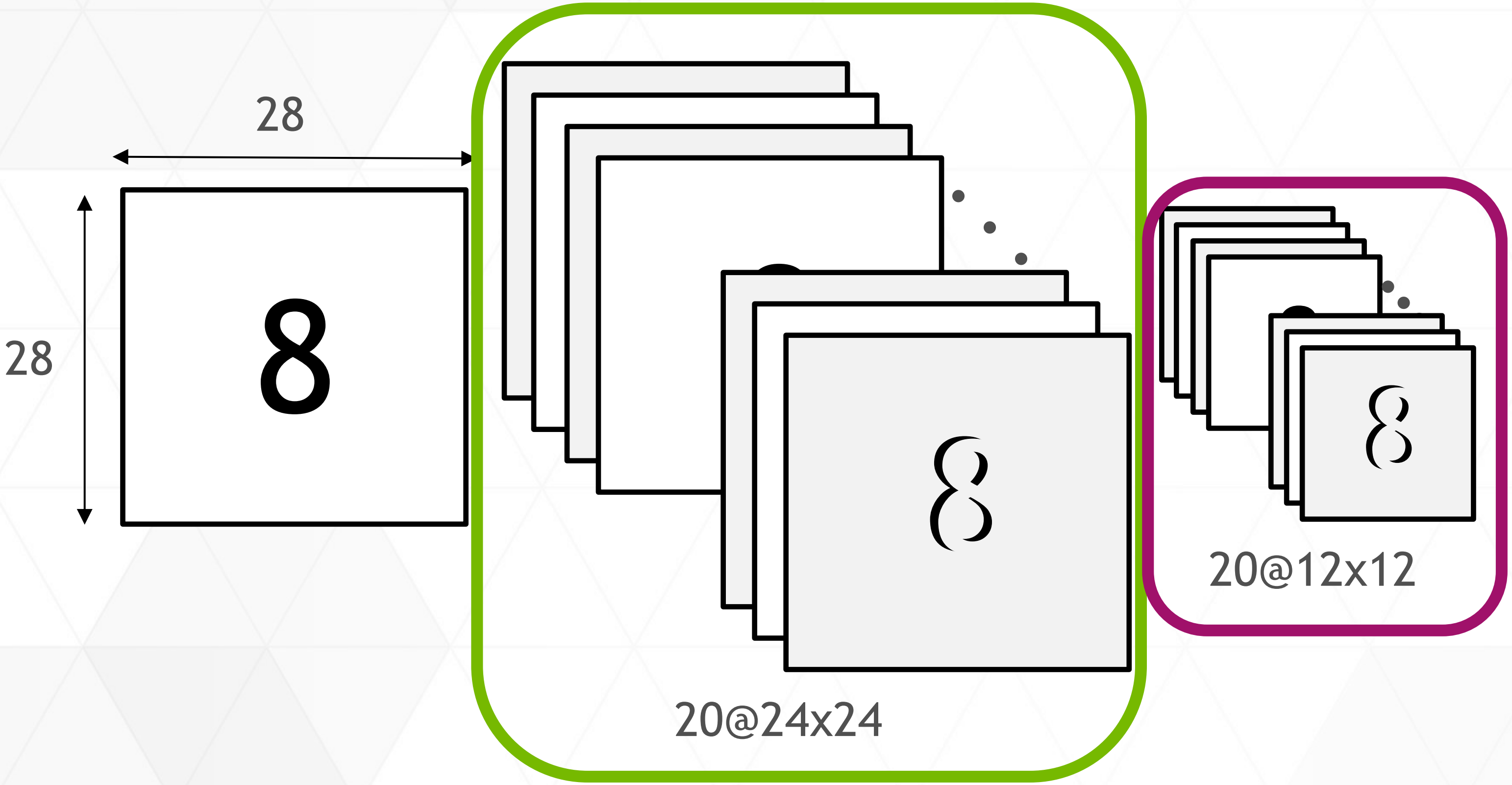
mnist_28x28
Done 05:03:02 PM
Image Size
28x28
Image Type
COLOR
Create DB (train)
45002 images
Create DB (val)
14998 images
Encoding

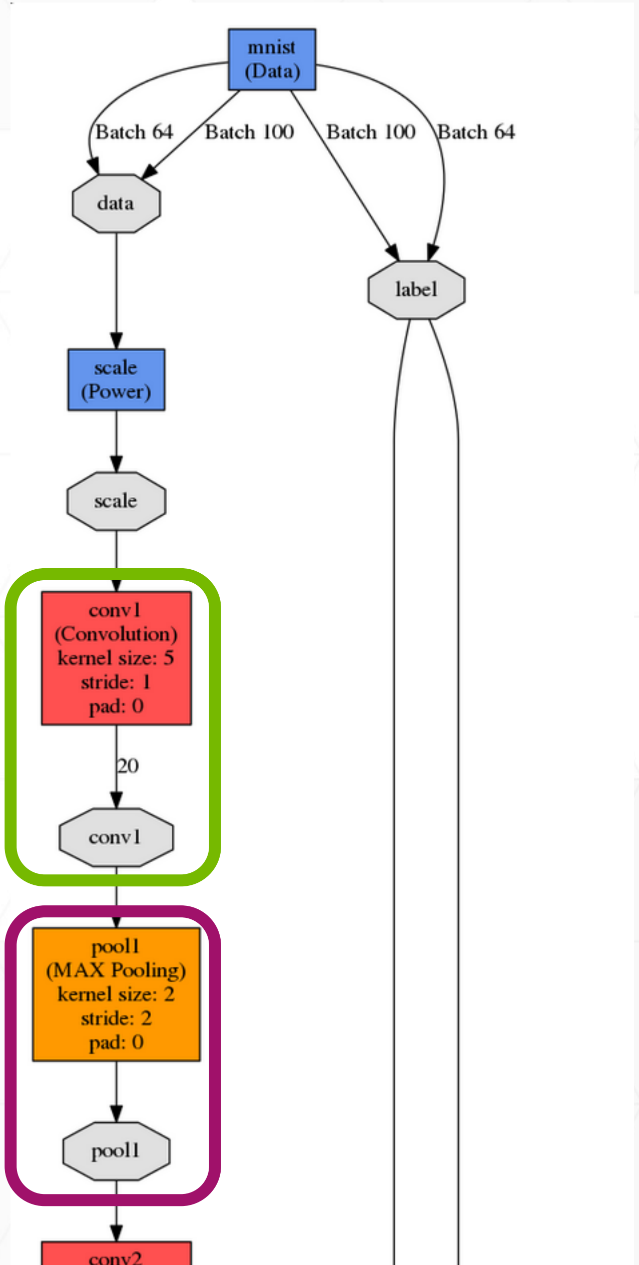
Data Transformations
Crop Size: none
Subtract Mean File: Yes

Solver Options
Training epochs: 30
Snapshot interval (in epochs): 1
Validation interval (in epochs): 1
Random seed: [none]
Batch size: [network defaults]
Solver type: Stochastic gradient descent (SGD)
Base Learning Rate: 0.01
□ Show advanced learning rate options

Model Name

NETWORK PARAMETERS

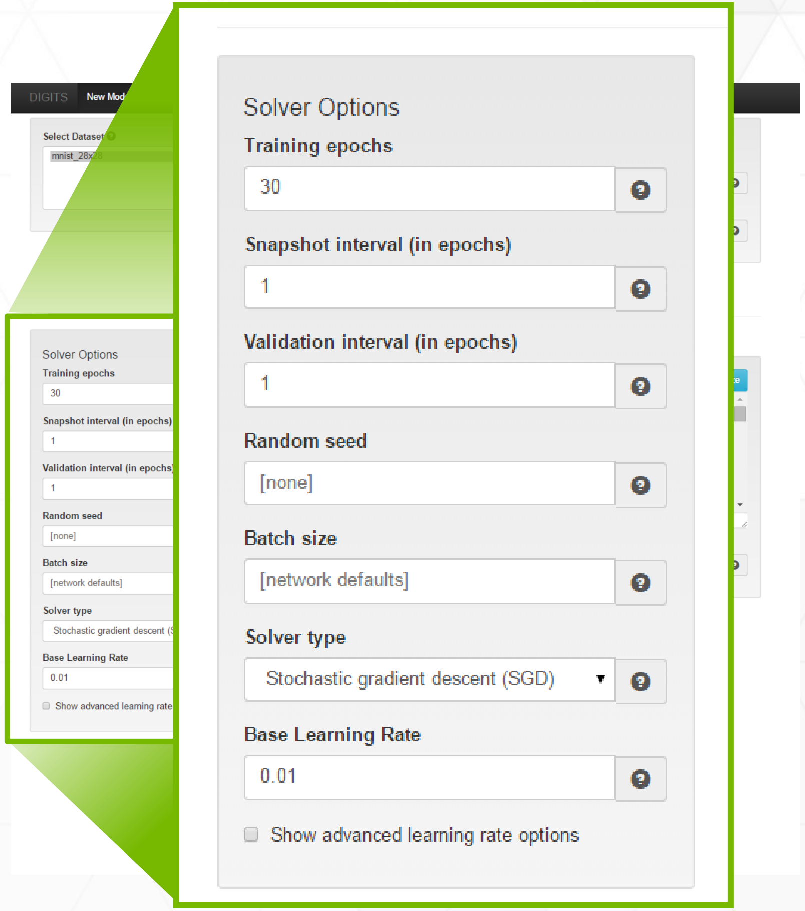




NVIDIA DIGITS

Solver Parameters

- Training epochs – processing of all data
- Snapshot interval – saving trained network
- Validation interval – DNN test with the validation data
- Batch size – number of images processed together
- Solver type – SGD, ADAGRAD,NAG
- Learning rate and policy



Solver Options

Training epochs
30

Snapshot interval (in epochs)
1

Validation interval (in epochs)
1

Random seed
[none]

Batch size
[network defaults]

Solver type
Stochastic gradient descent (SGD) ▾

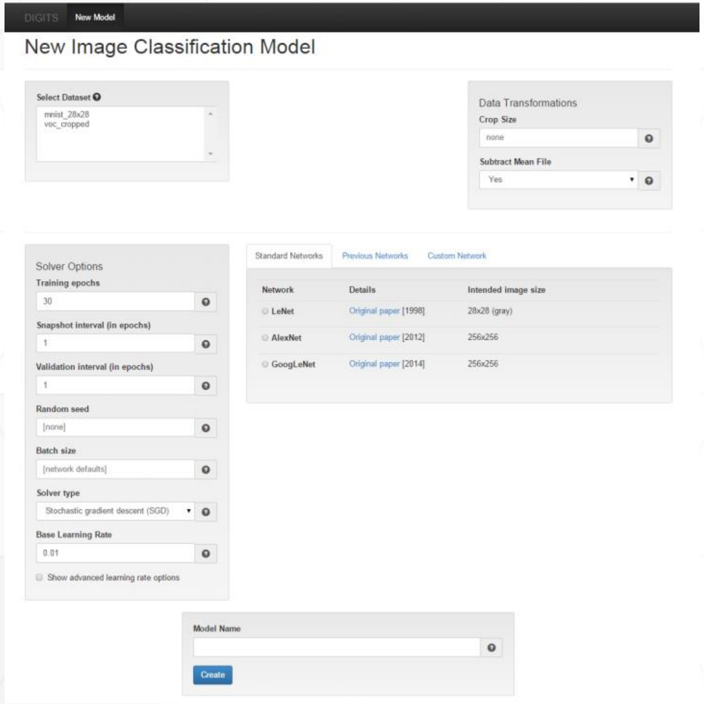
Base Learning Rate
0.01

□ Show advanced learning rate options

NVIDIA DIGITS

Single and multi-GPU training is easy

Singe GPU system



New Image Classification Model

Select Dataset
mnist_28x28
voc_cropped

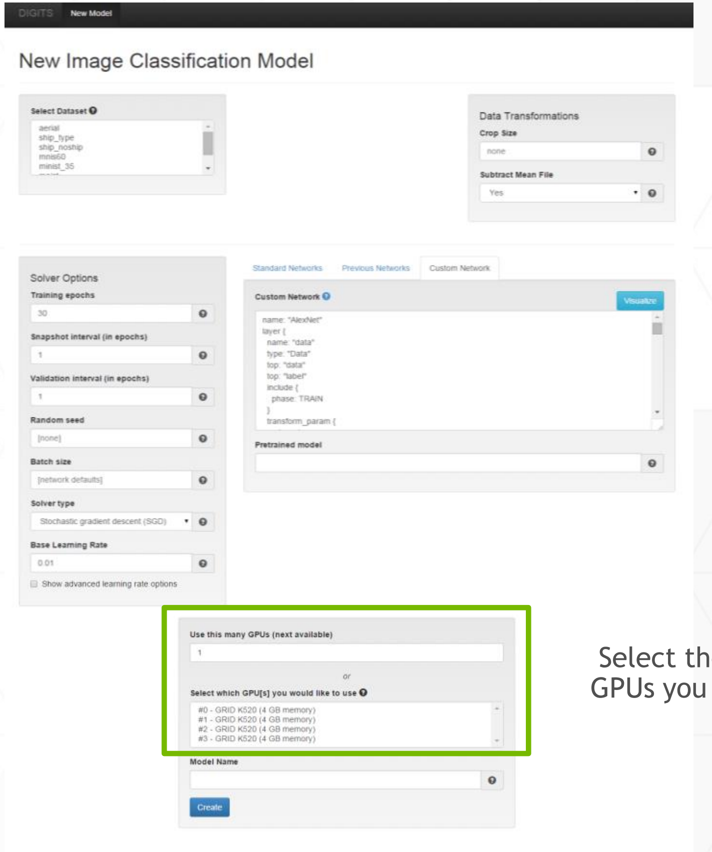
Data Transformations
Crop Size: none
Subtract Mean File: Yes

Solver Options
Training epochs: 30
Snapshot interval (in epochs): 1
Validation interval (in epochs): 1
Random seed: [none]
Batch size: [network defaults]
Solver type: Stochastic gradient descent (SGD)
Base Learning Rate: 0.01
□ Show advanced learning rate options

Network	Details	Intended image size
○ LeNet	Original paper [1998]	28x28 (gray)
○ AlexNet	Original paper [2012]	256x256
○ GoogLeNet	Original paper [2014]	256x256

Model Name

Multiple GPU system



New Image Classification Model

Select Dataset
aerial
ship_type
ship_noship
mnis60
minist_35

Data Transformations
Crop Size: none
Subtract Mean File: Yes

Solver Options
Training epochs: 30
Snapshot interval (in epochs): 1
Validation interval (in epochs): 1
Random seed: [none]
Batch size: [network defaults]
Solver type: Stochastic gradient descent (SGD)
Base Learning Rate: 0.01
□ Show advanced learning rate options

Custom Network

```
name: "AlexNet"
layer {
  name: "data"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TRAIN
  }
  transform_param {
```

Pretrained model

Use this many GPUs (next available): 1

or

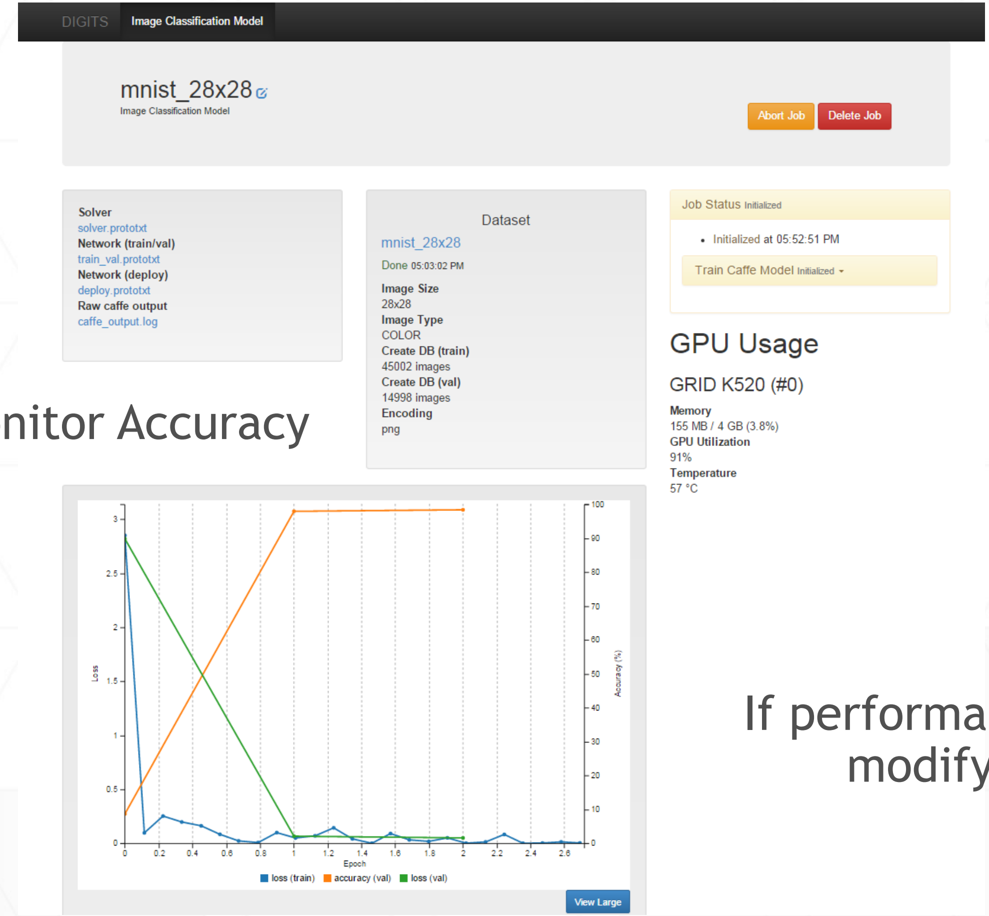
Select which GPU[s] you would like to use
#0 - GRID K520 (4 GB memory)
#1 - GRID K520 (4 GB memory)
#2 - GRID K520 (4 GB memory)
#3 - GRID K520 (4 GB memory)

Model Name

Select the number of GPUs you want to use.

NVIDIA DIGITS

Training Results



mnist_28x28

Image Classification Model

Solver
solver.prototxt
Network (train/val)
train_val.prototxt
Network (deploy)
deploy.prototxt
Raw caffe output
caffe_output.log

Dataset

mnist_28x28

Done 05:03:02 PM

Image Size
28x28
Image Type
COLOR
Create DB (train)
45002 images
Create DB (val)
14998 images
Encoding
png

Job Status Initialized

- Initialized at 05:52:51 PM

Train Caffe Model Initialized

GPU Usage

GRID K520 (#0)

Memory
155 MB / 4 GB (3.8%)
GPU Utilization
91%
Temperature
57 °C

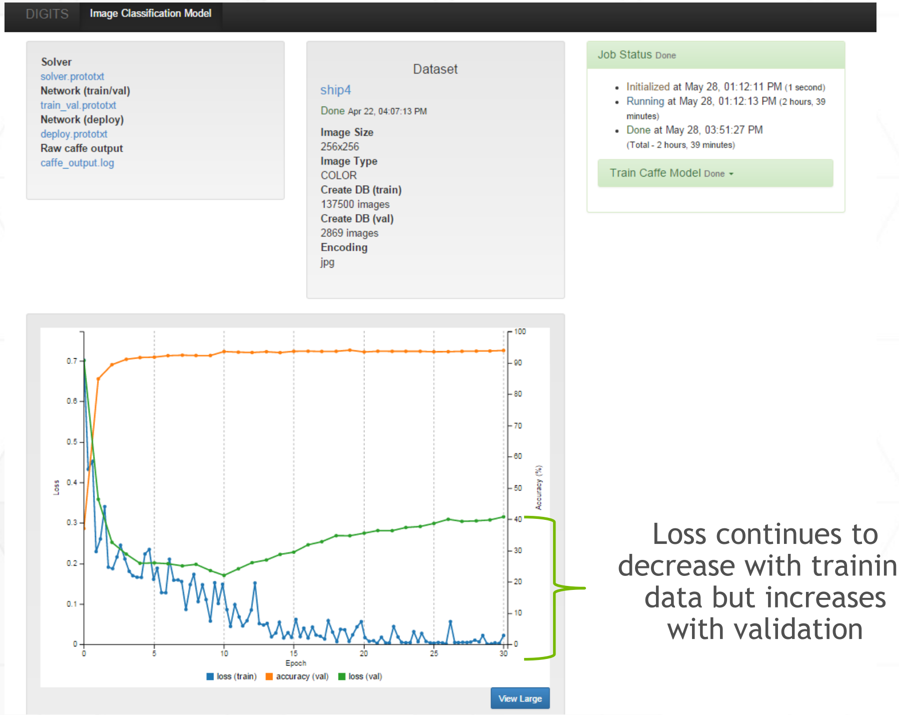
Monitor Accuracy

If performance is poor, abort, modify, and retrain.

OVERFITTING AND UNDERFITTING

How can I use DIGITS to tell me this is happening?

Overfitting



DIGITS | Image Classification Model

Solver
solver.prototxt
Network (train/val)
train_val.prototxt
Network (deploy)
deploy.prototxt
Raw caffe output
caffe_output.log

Dataset

ship4
Done Apr 22, 04:07:13 PM
Image Size
256x256
Image Type
COLOR
Create DB (train)
137500 images
Create DB (val)
2869 images
Encoding
jpg

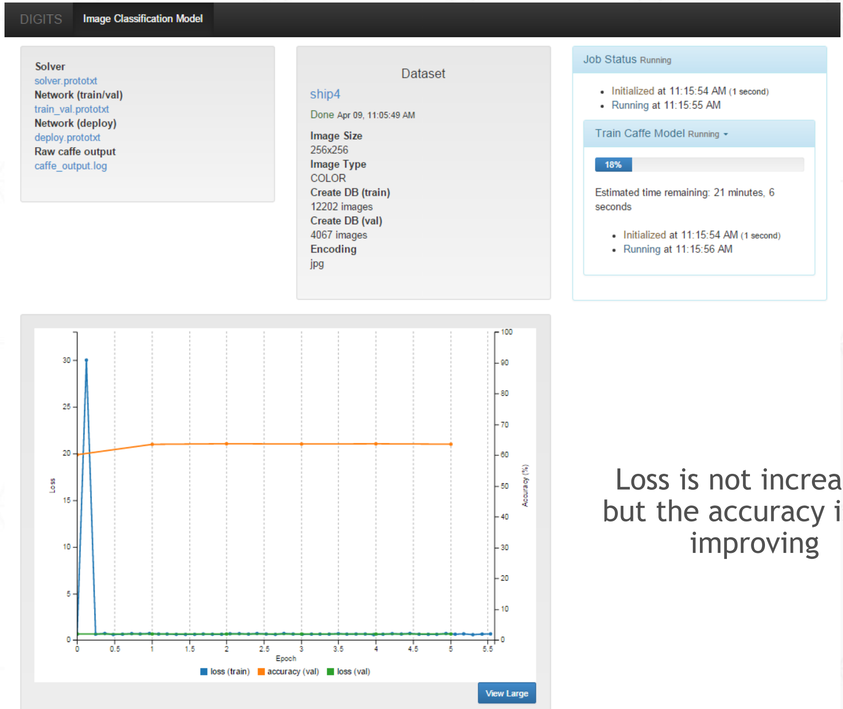
Job Status Done

- Initialized at May 28, 01:12:11 PM (1 second)
- Running at May 28, 01:12:13 PM (2 hours, 39 minutes)
- Done at May 28, 03:51:27 PM (Total - 2 hours, 39 minutes)

Train Caffe Model Done

Loss continues to decrease with training data but increases with validation

Underfitting



DIGITS | Image Classification Model

Solver
solver.prototxt
Network (train/val)
train_val.prototxt
Network (deploy)
deploy.prototxt
Raw caffe output
caffe_output.log

Dataset

ship4
Done Apr 09, 11:05:49 AM
Image Size
256x256
Image Type
COLOR
Create DB (train)
12202 images
Create DB (val)
4067 images
Encoding
jpg

Job Status Running

- Initialized at 11:15:54 AM (1 second)
- Running at 11:15:55 AM

Train Caffe Model Running

18%

Estimated time remaining: 21 minutes, 6 seconds

- Initialized at 11:15:54 AM (1 second)
- Running at 11:15:56 AM

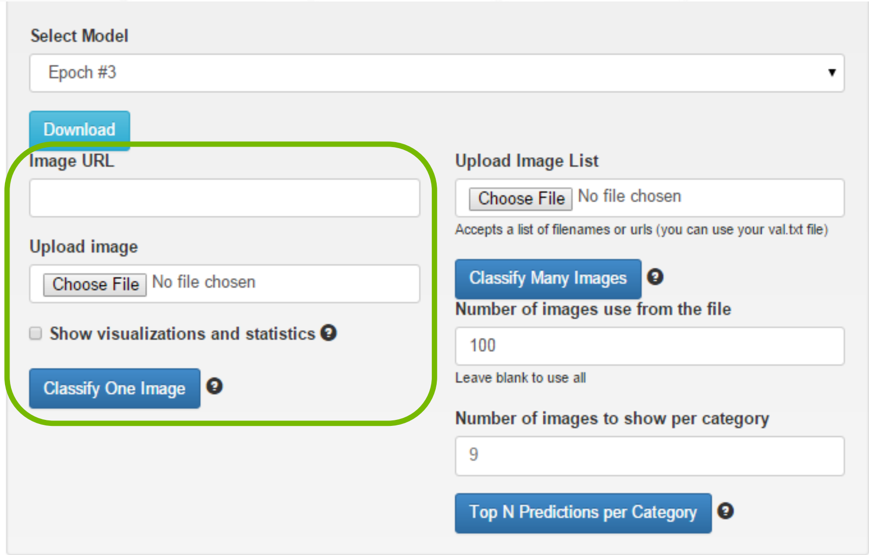
Loss is not increasing but the accuracy is not improving

Validation data helps you identify when/if this occurs!

NVIDIA DIGITS

Single Image Classification

*



Select Model: Epoch #3

Image URL:

Upload image: Choose File No file chosen

□ Show visualizations and statistics

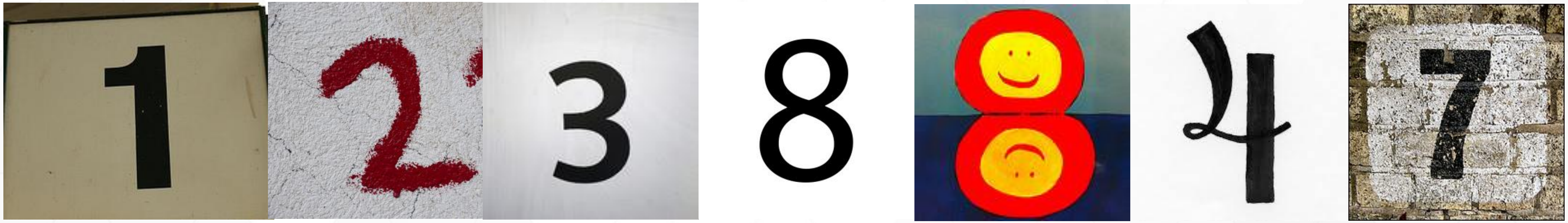
Upload Image List: Choose File No file chosen

Accepts a list of filenames or urls (you can use your val.txt file)

Number of images use from the file: 100

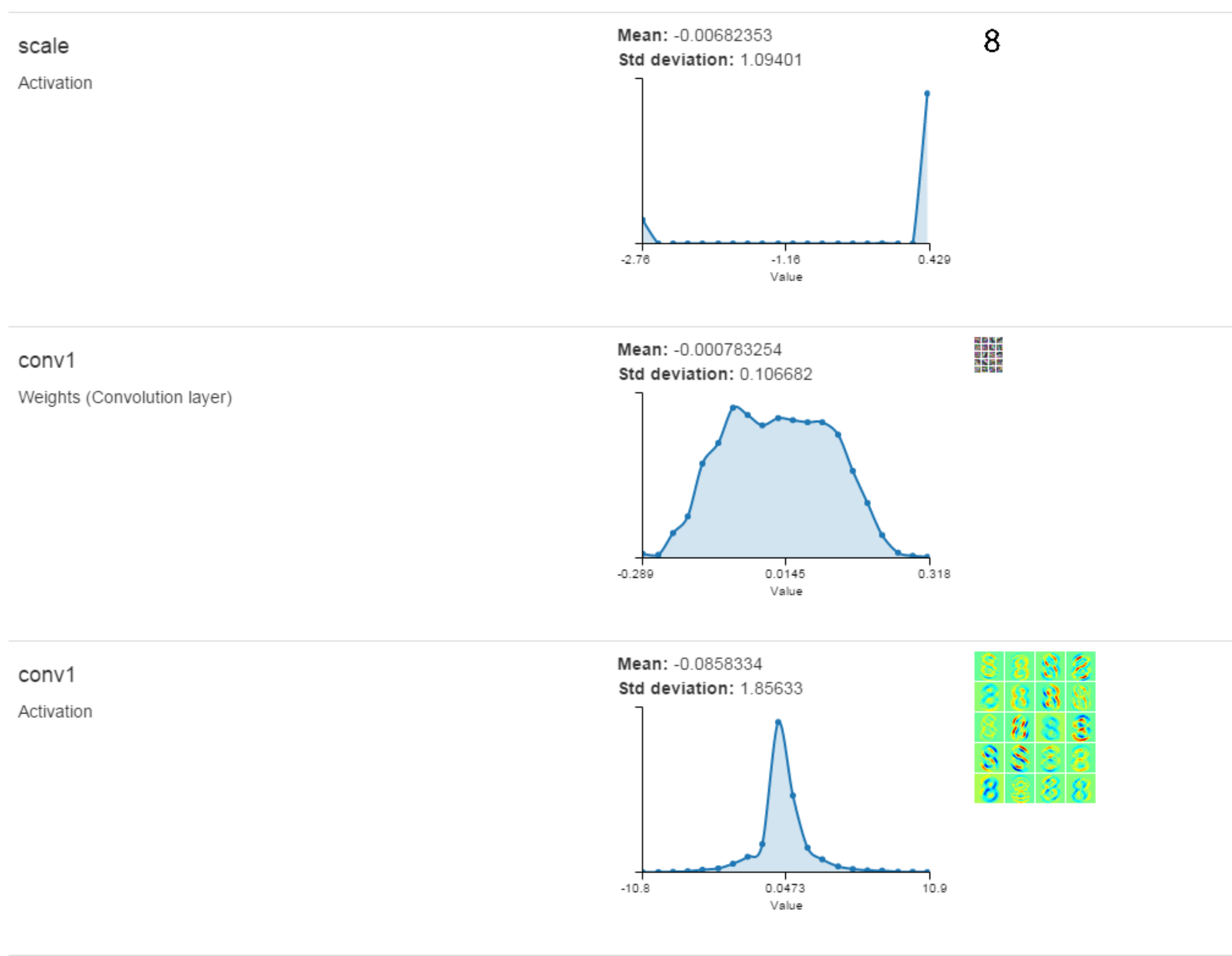
Leave blank to use all

Number of images to show per category: 9



* Lab tasks

SINGLE IMAGE CLASSIFICATION RESULTS



scale

Activation

Mean: -0.00682353

Std deviation: 1.09401

conv1

Weights (Convolution layer)

Mean: -0.000783254

Std deviation: 0.106682

conv1

Activation

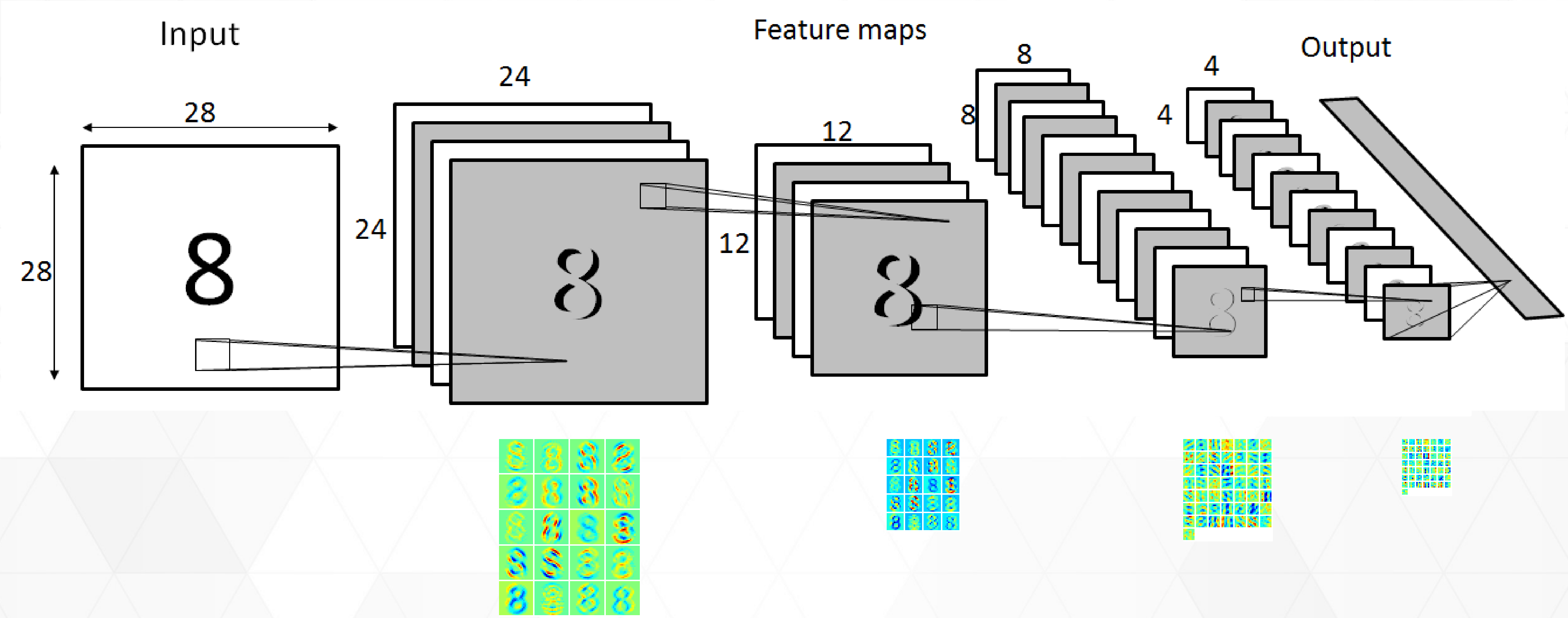
Mean: -0.0858334

Std deviation: 1.85633

- Network response at each layer will display
- Visualize responses from different inputs

NETWORK CONFIGURATION

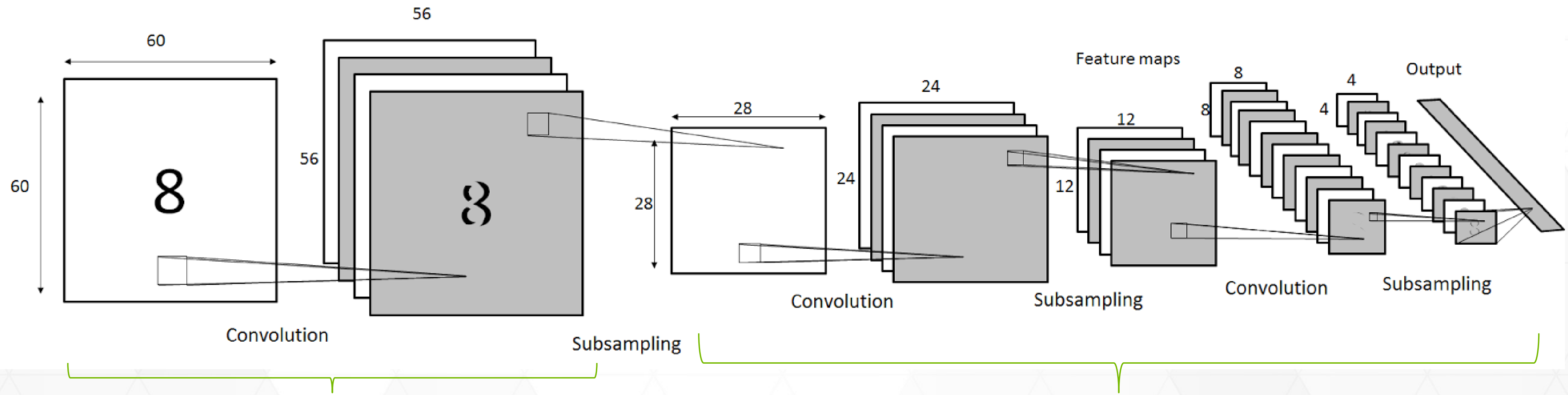
Visualizing LeNet Network Responses



NETWORK CONFIGURATIONS

Modifying a Network

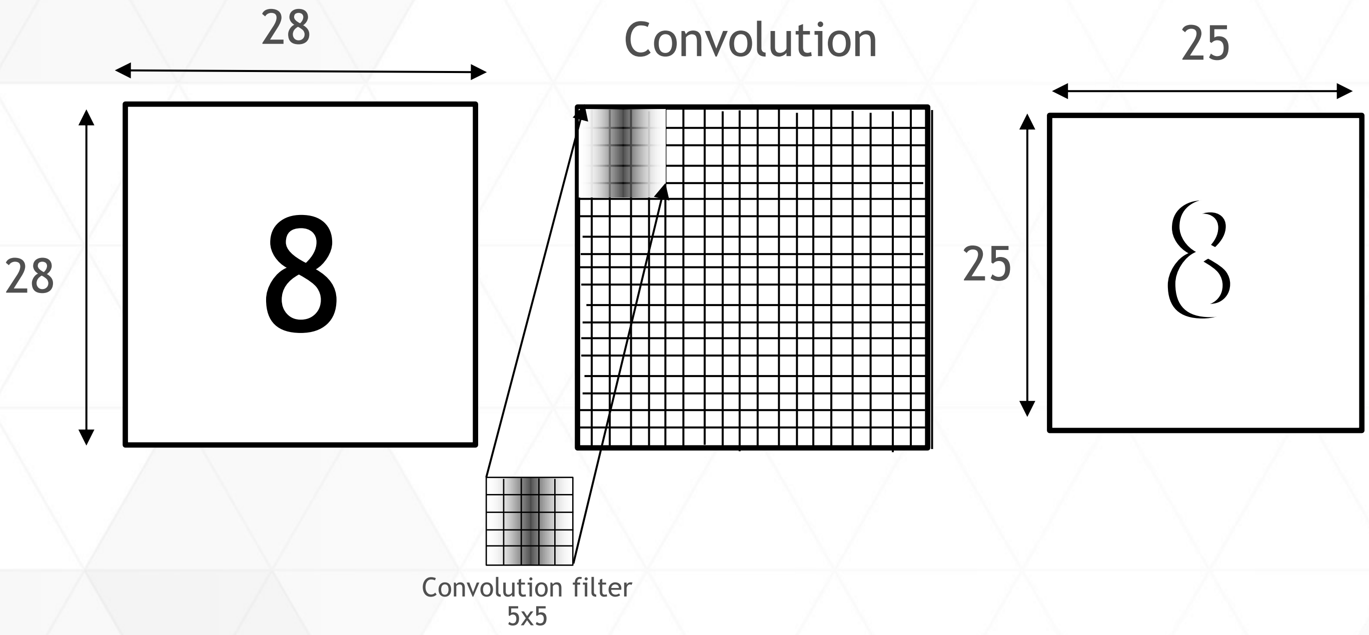
- Modifying a network can improve performance
- There are many parameters - add or remove a layer, pooling, activation function, zero padding, increasing outputs



Modification – added a convolutional layer and pooling

LetNet

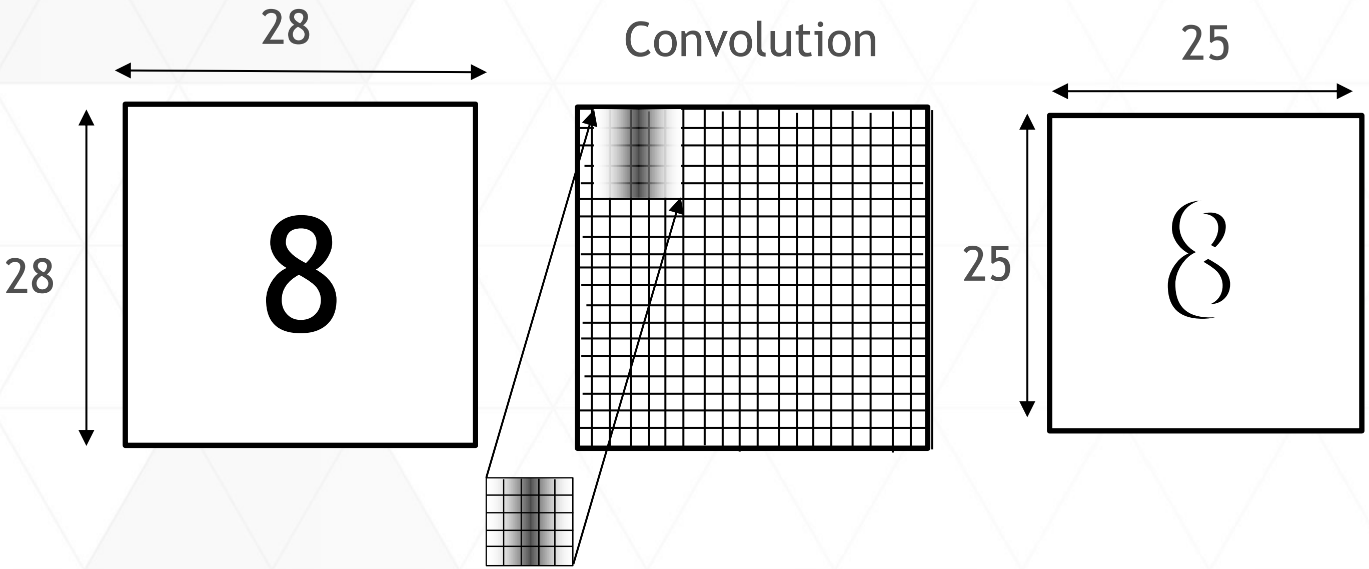
MODIFYING YOUR NETWORK



Zero padding, input is reduced by the radius of the kernel
Input[28x28]*filter[5x5]=FeatureMap[25x25]

```
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "scale"
  top: "conv1"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }
  convolution_param {
    num_output: 20
    kernel_size: 5
    stride: 1
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
```

MODIFYING YOUR NETWORK

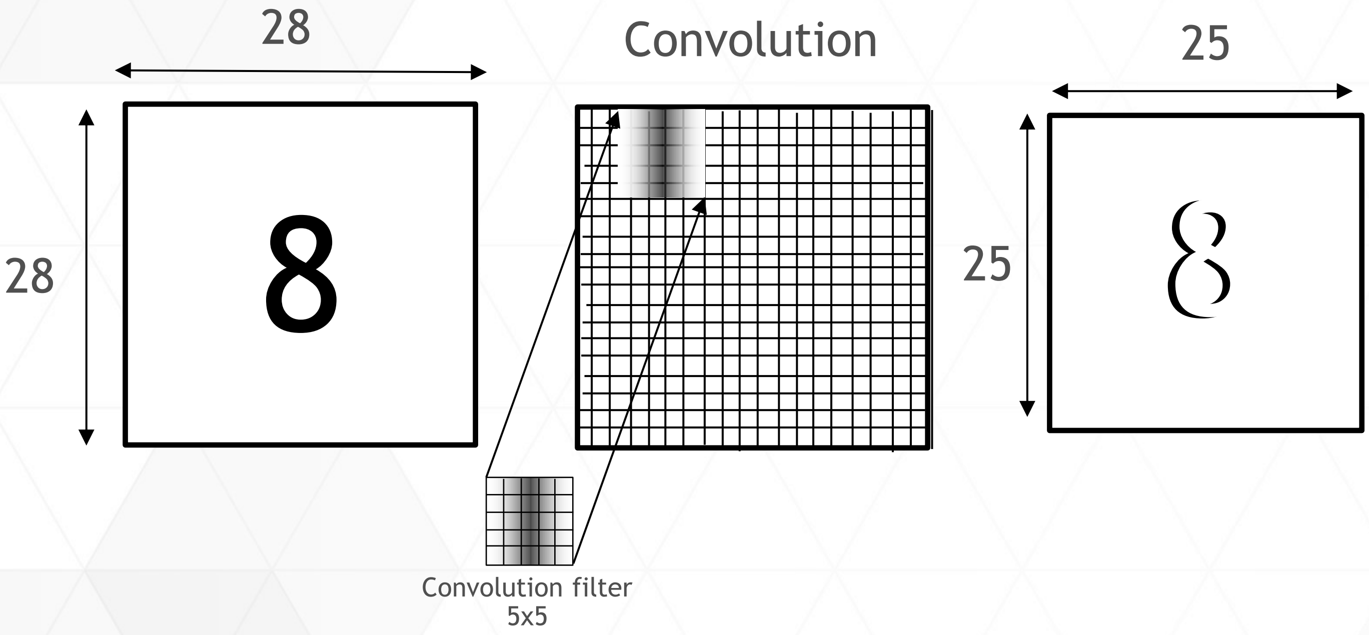


Convolution filter 5x5

Zero padding, input is reduced by the radius of the kernel
Input[28x28]*filter[5x5]=FeatureMap[25x25]

```
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "scale"
  top: "conv1"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }
  convolution_param {
    num_output: 20
    kernel_size: 5
    stride: 1
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
```

MODIFYING YOUR NETWORK

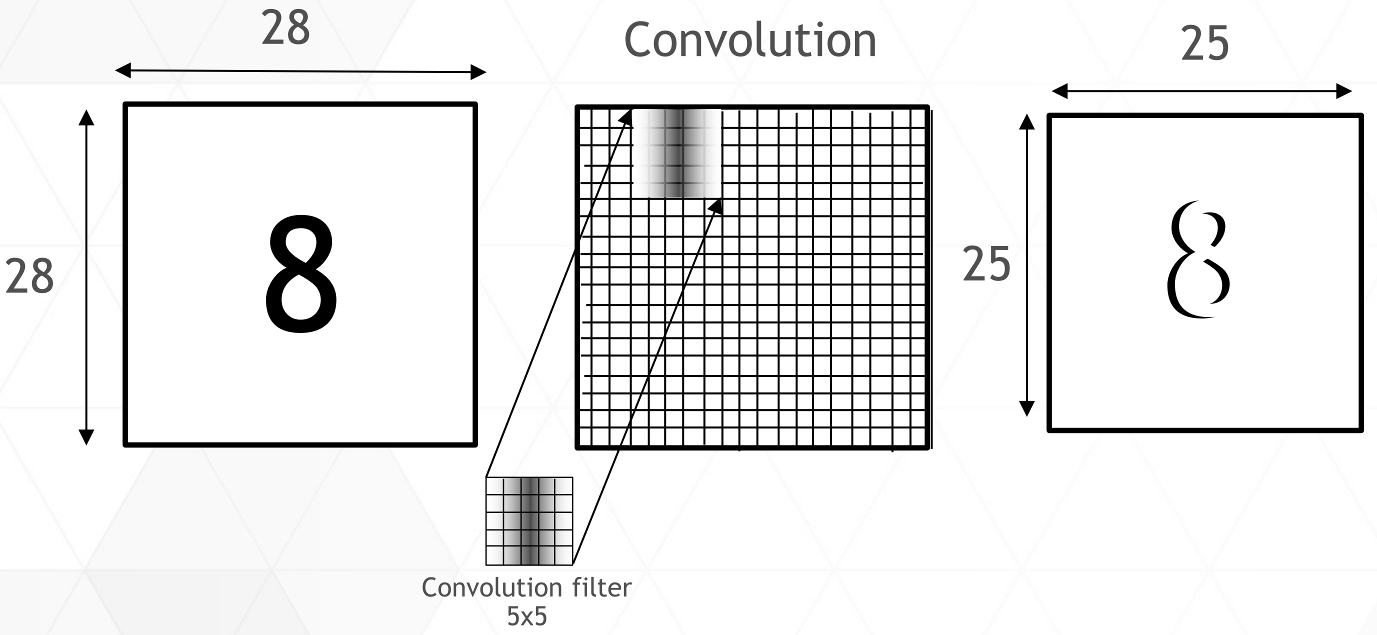


Convolution filter
5x5

Zero padding, input is reduced by the radius of the kernel
Input[28x28]*filter[5x5]=FeatureMap[25x25]

```
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "scale"
  top: "conv1"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }
  convolution_param {
    num_output: 20
    kernel_size: 5
    stride: 1
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
```

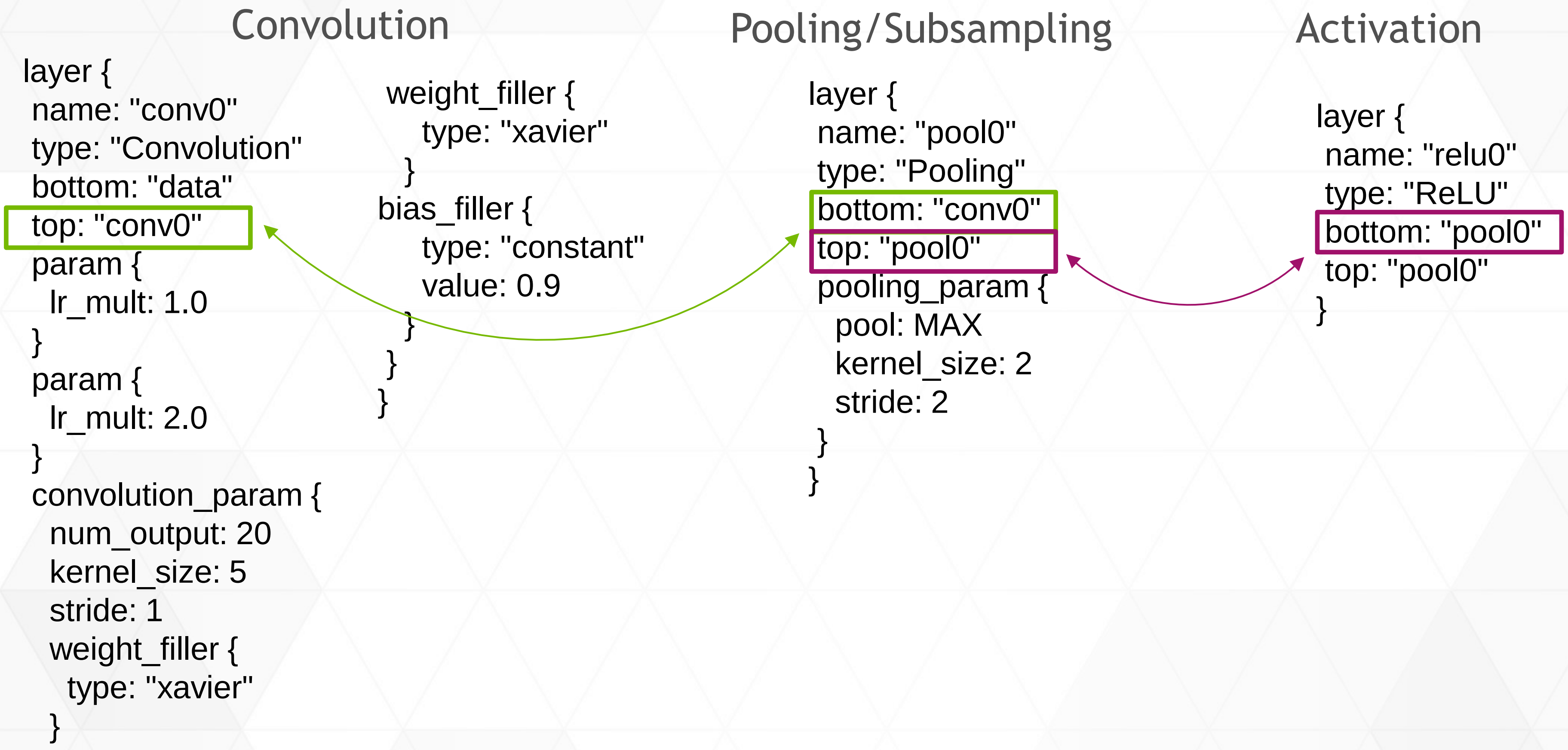
MODIFYING YOUR NETWORK



Zero padding, input is reduced by the radius of the kernel
Input[28x28]*filter[5x5]=FeatureMap[25x25]

```
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "scale"
  top: "conv1"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }
  convolution_param {
    num_output: 20
    kernel_size: 5
    stride: 1
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
```

NETWORK PARAMETERS



Convolution

```
layer {
 name: "conv0"
 type: "Convolution"
 bottom: "data"
 top: "conv0"
 param {
   lr_mult: 1.0
 }
 param {
   lr_mult: 2.0
 }
 convolution_param {
   num_output: 20
   kernel_size: 5
   stride: 1
   weight_filler {
     type: "xavier"
   }
```

```
  weight_filler {
     type: "xavier"
   }
 bias_filler {
     type: "constant"
     value: 0.9
   }
  }
}
```

Pooling/Subsampling

```
layer {
 name: "pool0"
 type: "Pooling"
 bottom: "conv0"
 top: "pool0"
 pooling_param {
   pool: MAX
   kernel_size: 2
   stride: 2
 }
}
```

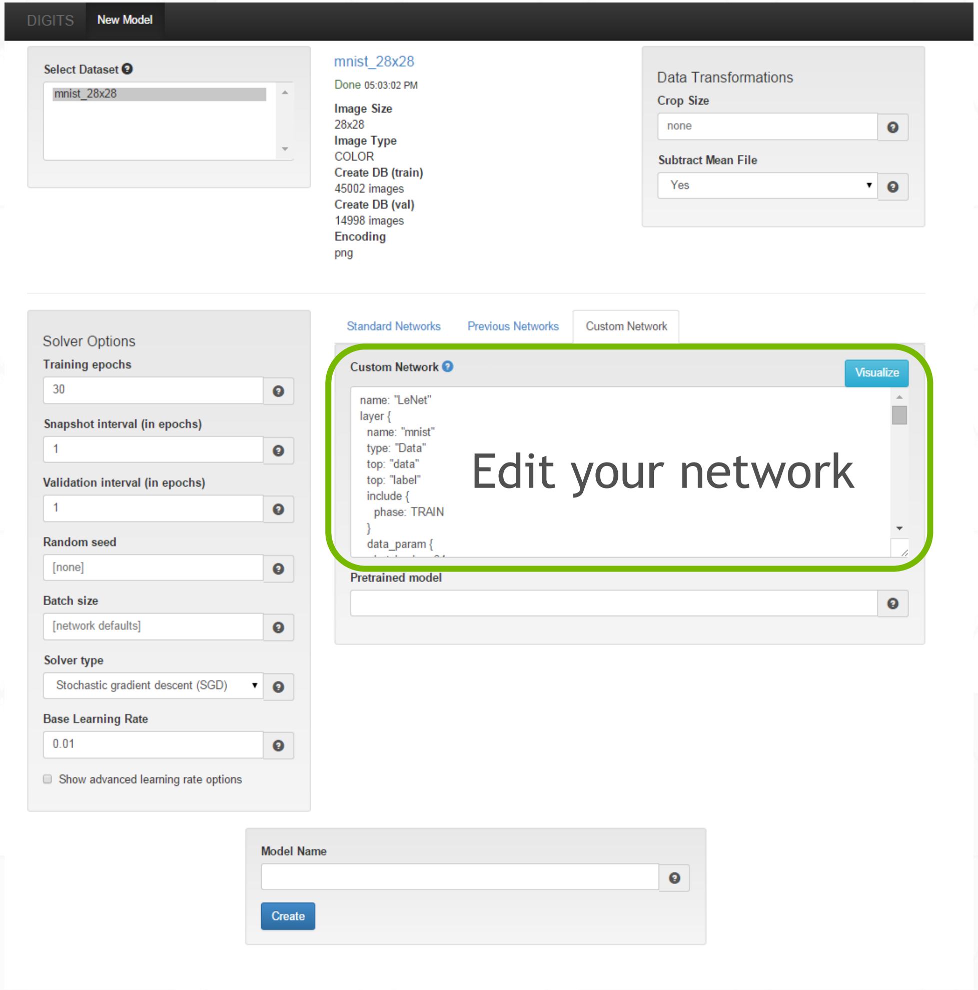
Activation

```
layer {
 name: "relu0"
 type: "ReLU"
 bottom: "pool0"
 top: "pool0"
}
```

* Lab tasks

NVIDIA DIGITS

Modifying your Network



Select Dataset
mnist_28x28

mnist_28x28
Done 05:03:02 PM

Image Size
28x28
Image Type
COLOR
Create DB (train)
45002 images
Create DB (val)
14998 images
Encoding
png

Data Transformations
Crop Size: none
Subtract Mean File: Yes

Solver Options
Training epochs: 30
Snapshot interval (in epochs): 1
Validation interval (in epochs): 1
Random seed: [none]
Batch size: [network defaults]
Solver type: Stochastic gradient descent (SGD)
Base Learning Rate: 0.01
□ Show advanced learning rate options

Custom Network

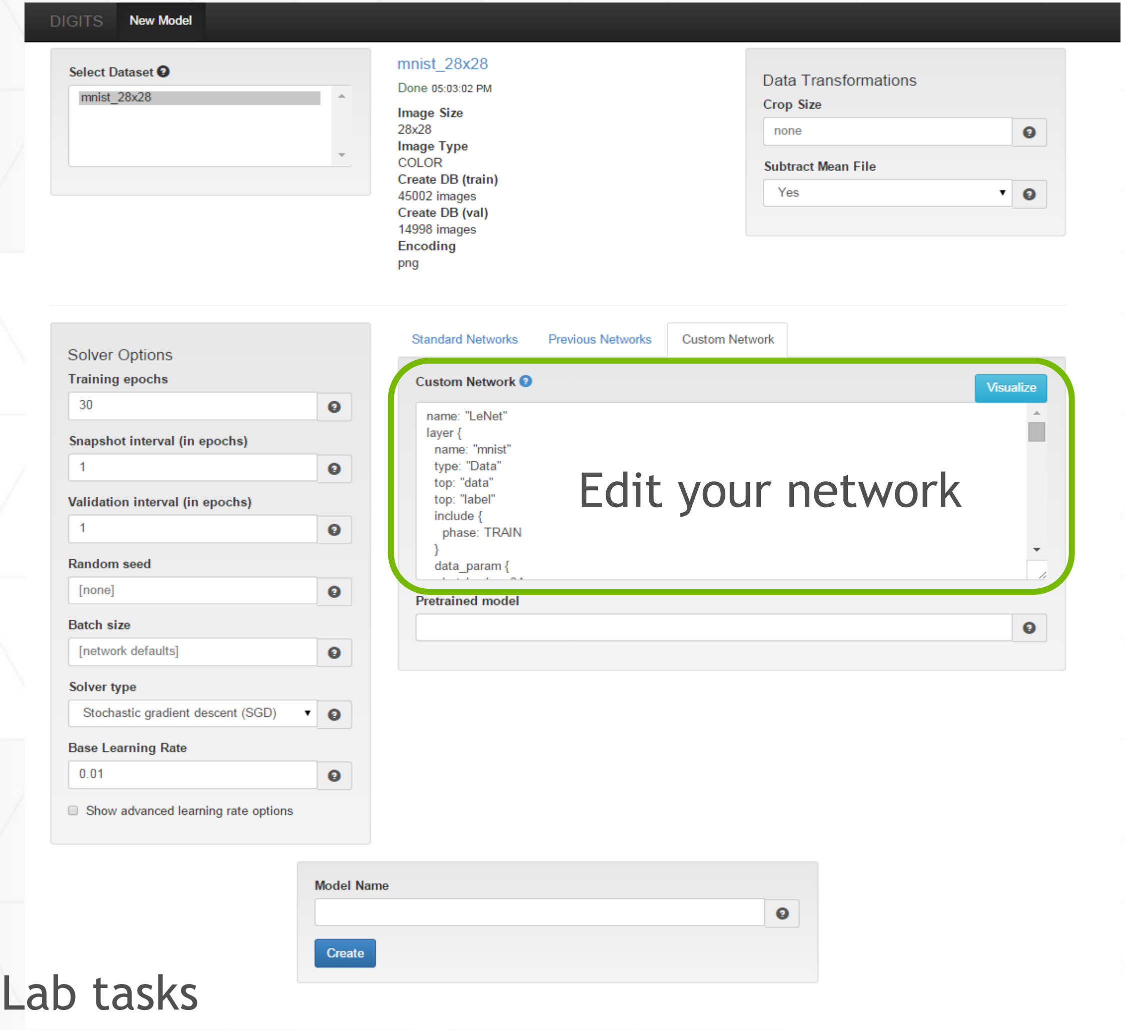
```
name: "LeNet"
layer {
  name: "mnist"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TRAIN
  }
  data_param {
```

Edit your network

Pretrained model

Model Name

MODIFYING YOUR NETWORK



Select Dataset: mnist_28x28

mnist_28x28
Done 05:03:02 PM
Image Size
28x28
Image Type
COLOR
Create DB (train)
45002 images
Create DB (val)
14998 images
Encoding
png

Data Transformations
Crop Size: none
Subtract Mean File: Yes

Solver Options
Training epochs: 30
Snapshot interval (in epochs): 1
Validation interval (in epochs): 1
Random seed: [none]
Batch size: [network defaults]
Solver type: Stochastic gradient descent (SGD)
Base Learning Rate: 0.01
□ Show advanced learning rate options

Custom Network

```
name: "LeNet"
layer {
  name: "mnist"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TRAIN
  }
  data_param {
```

Edit your network

Pretrained model

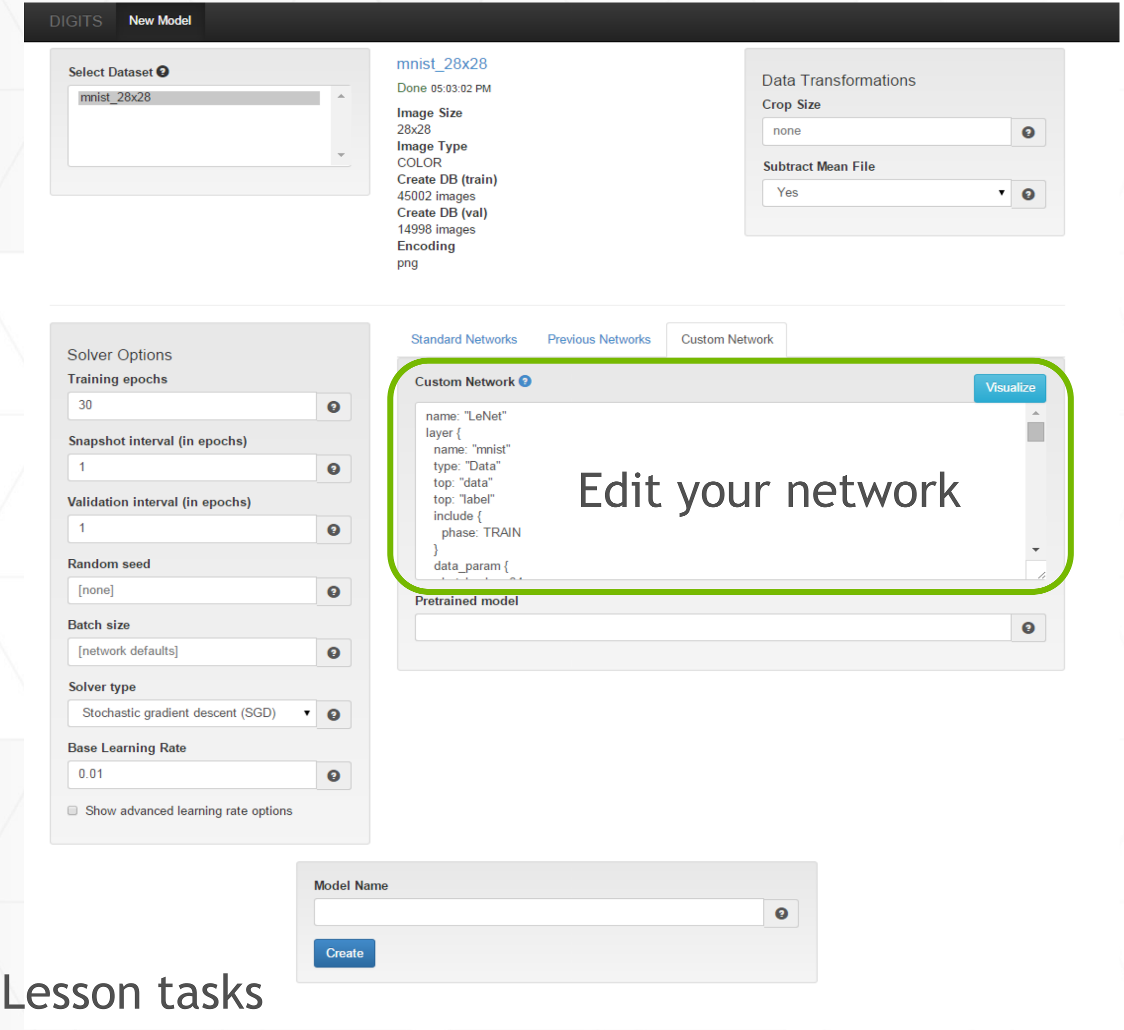
Model Name

```
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "scale"
  top: "conv1"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }
  convolution_param {
    num_output: 20   *
    kernel_size: 5
    stride: 1
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
```

Input and output to the layer

* Lab tasks

MODIFYING YOUR NETWORK



Select Dataset
mnist_28x28

mnist_28x28
Done 05:03:02 PM
Image Size
28x28
Image Type
COLOR
Create DB (train)
45002 images
Create DB (val)
14998 images
Encoding
png

Data Transformations
Crop Size: none
Subtract Mean File: Yes

Solver Options
Training epochs: 30
Snapshot interval (in epochs): 1
Validation interval (in epochs): 1
Random seed: [none]
Batch size: [network defaults]
Solver type: Stochastic gradient descent (SGD)
Base Learning Rate: 0.01
□ Show advanced learning rate options

Custom Network

```
name: "LeNet"
layer {
  name: "mnist"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TRAIN
  }
  data_param {
```

Edit your network

Pretrained model

Model Name

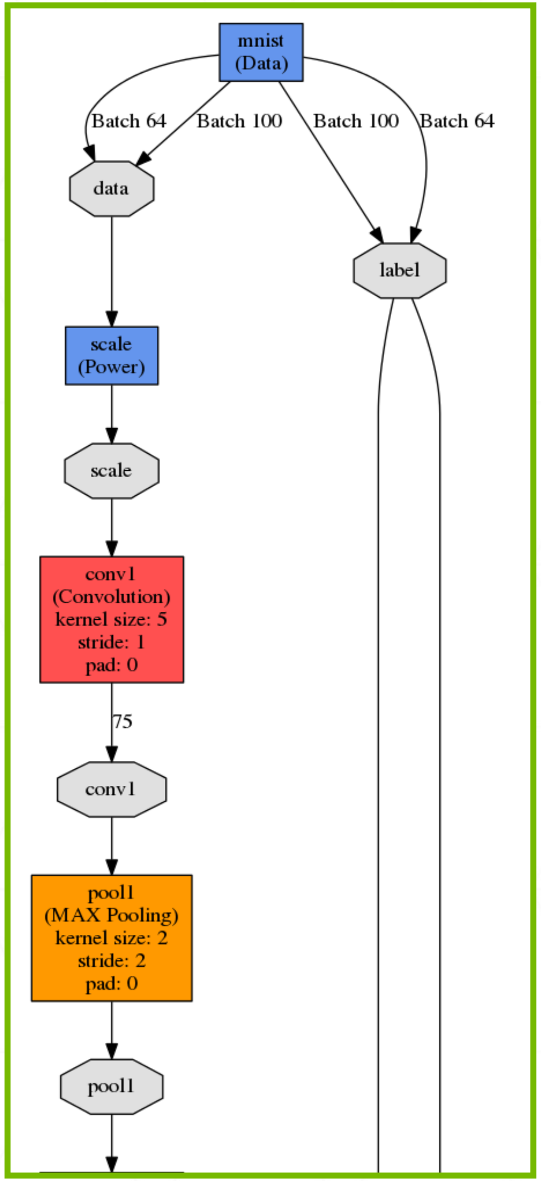
```
* layer {
    name: "relu1"
    type: "ReLU"
    bottom: "conv1"
    top: "conv1"
  }
```

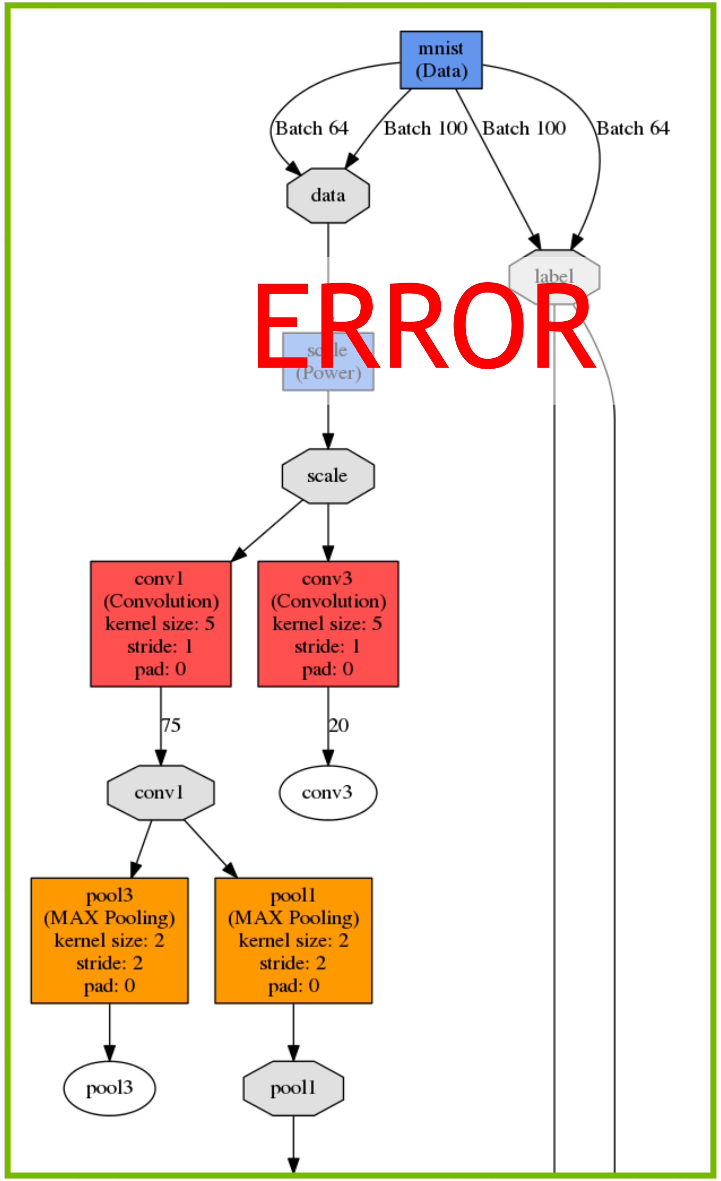
Input and output to the layer

* Lesson tasks

MODIFYING YOU NETWORK

Visualize Configuration Changes

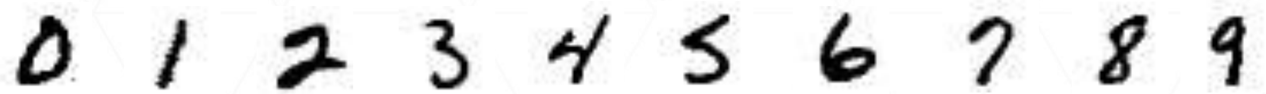

mnist (Data)
Batch 64
Batch 100
Batch 100
Batch 64
data
label
scale (Power)
scale
conv1 (Convolution) kernel size: 5 stride: 1 pad: 0
75
conv1
pool1 (MAX Pooling) kernel size: 2 stride: 2 pad: 0
pool1


mnist (Data)
Batch 64
Batch 100
Batch 100
Batch 64
data
label
ERROR
scale (Power)
scale
conv1 (Convolution) kernel size: 5 stride: 1 pad: 0
conv3 (Convolution) kernel size: 5 stride: 1 pad: 0
75
20
conv1
conv3
pool3 (MAX Pooling) kernel size: 2 stride: 2 pad: 0
pool1 (MAX Pooling) kernel size: 2 stride: 2 pad: 0
pool3
pool1

ANOTHER WAY TO IMPROVE PERFORMANCE

Data Augmentation

- Sometimes training data is not a great representation of the field data
    - MNIST data is grayscale, black text with white background

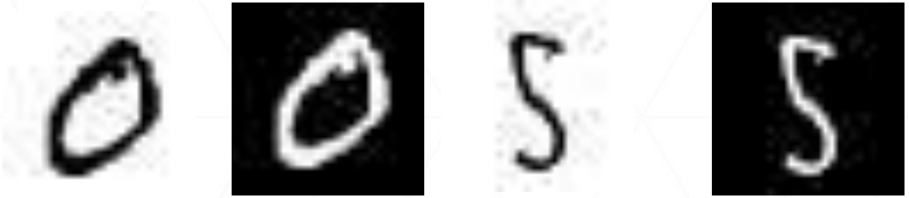


- Will these images to be classified correctly when the network is trained with this digit data?

ANOTHER WAY TO IMPROVE PERFORMANCE

Data Augmentation

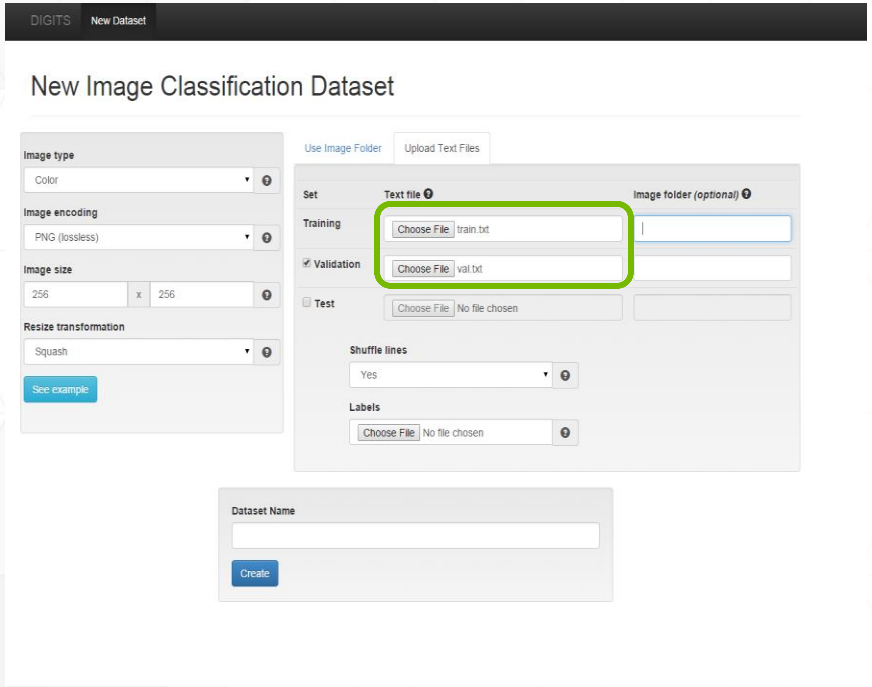
- Depending on the deployment scenario, simple modifications can be made to the training data to improve performance



- There are many ways to augment data
  - Rotations, noise, color distortions, stretching, etc. *
- Many ways to modify images – ImageMagick, Python Pillow, OpenCV

* Lesson tasks

USING AN AUGMENTED DATA SET



New Image Classification Dataset

Image type: Color

Image encoding: PNG (lossless)

Image size: 256 x 256

Resize transformation: Squash

Set	Text file	Image folder *(optional)*
Training	Choose File train.txt	
☑ Validation	Choose File val.txt	
□ Test	Choose File No file chosen	

Shuffle lines: Yes

Labels: Choose File No file chosen

Dataset Name

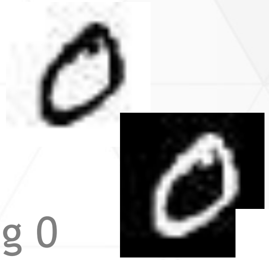
- train.txt

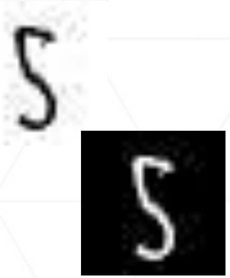
/home/user/train/0/0_1.jpg 0

/home/user/train/0/0_1_invert.jpg 0

/home/user/train/5/5_1.jpg 5

/home/user/train/5/5_1_invert.jpg 5

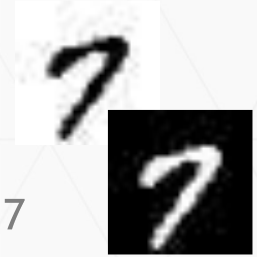




- val.txt

/home/user/mnist/val/7_1.jpg 7

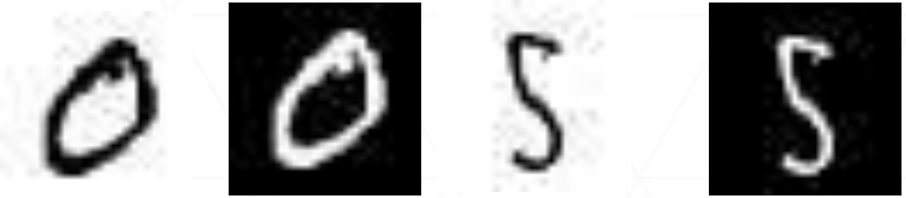
/home/user/mnist/val/7_1_invert.jpg 7



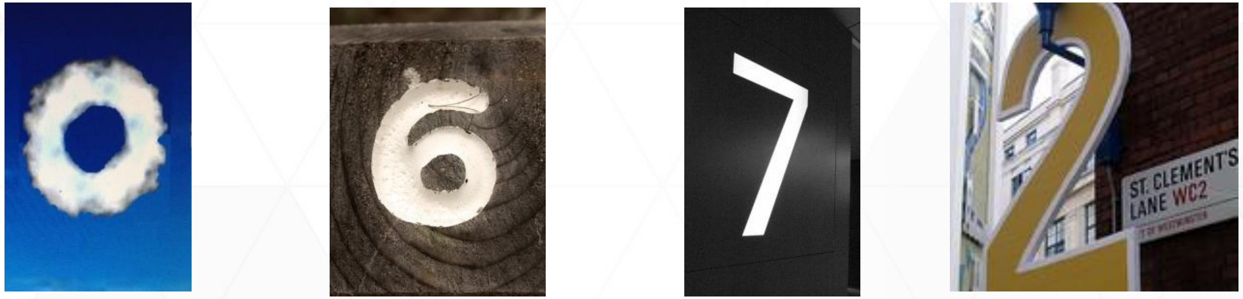
ANOTHER WAY TO IMPROVE PERFORMANCE

Data Augmentation

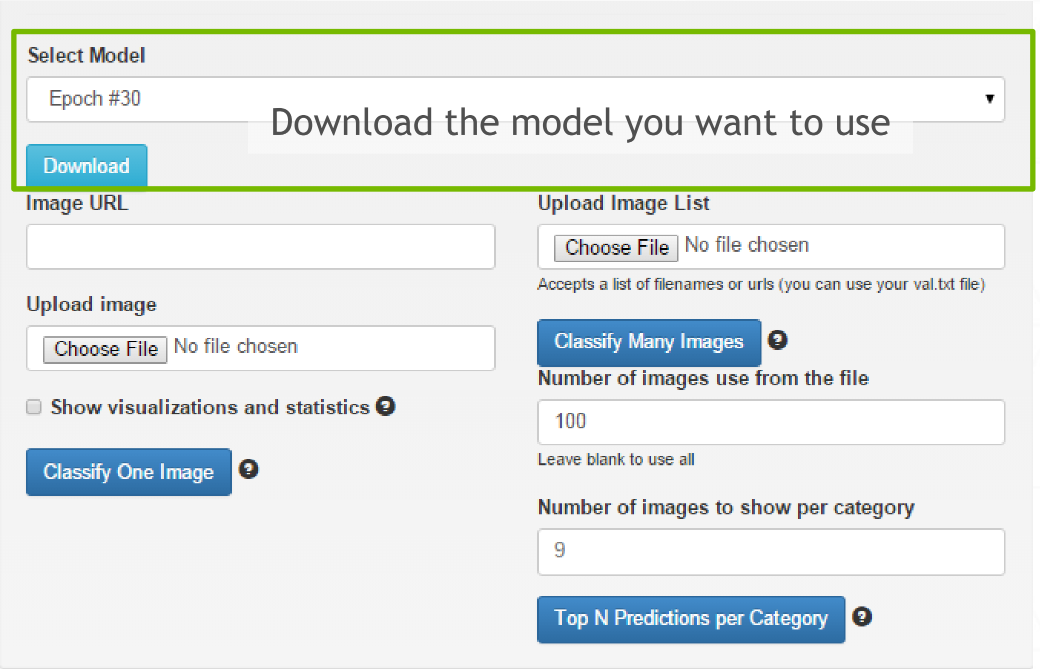
- Example augmentation - inverted copies of the input data



- Would a network trained with this data augmented, accurately classify these images?



DEPLOYING YOUR NETWORK



Select Model

Epoch #30 ▾

Download the model you want to use

Download

Image URL

Upload image

Choose File No file chosen

☐ Show visualizations and statistics

Classify One Image

Upload Image List

Choose File No file chosen

Accepts a list of filenames or urls (you can use your val.txt file)

Classify Many Images

Number of images use from the file

100

Leave blank to use all

Number of images to show per category

9

Top N Predictions per Category

- Deploy in the cloud

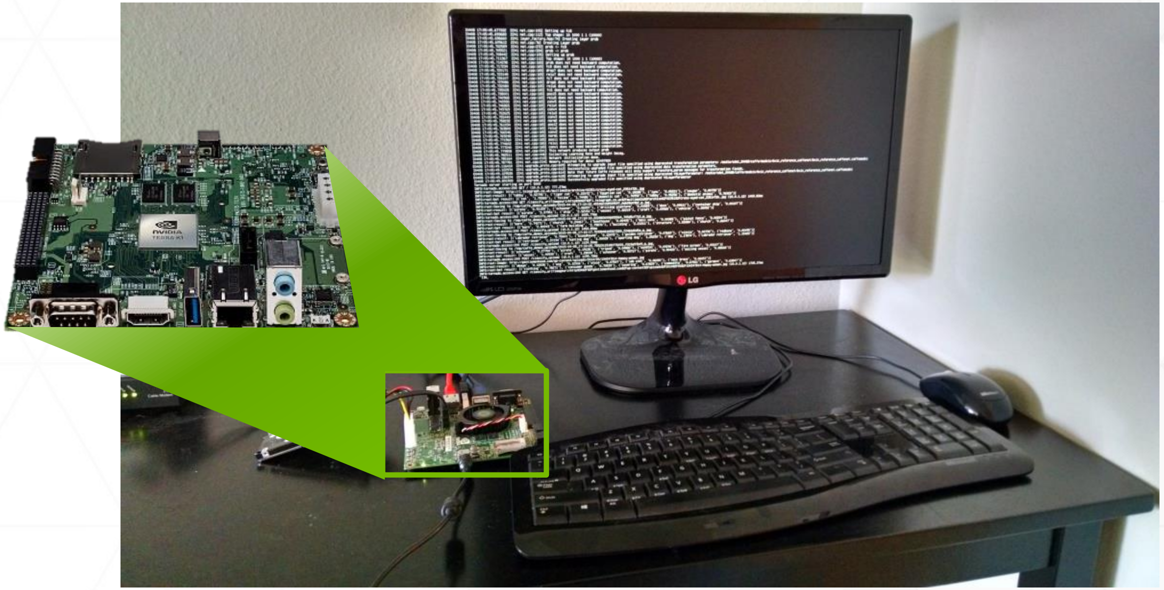
- Deploy on a mobile device

Tegra

DEPLOYMENT WITH TEGRA

Rapid Classification Anywhere

- ➢ Flexible
- ➢ Low Power
- ➢ Easy to use
- ➢ GPU accelerated



Build Caffe on your portable platform
Download your trained network from DIGITS

NVIDIA DIGITS

Resources

- Where to get DIGITS
  - Easy to use web installer https://developer.nvidia.com/digits
  - github - https://github.com/NVIDIA/DIGITS
    - Remember to install NVIDIA's Caffe branch - https://github.com/NVIDIA/caffe
- User support
  - DIGITS Users Google group - https://groups.google.com/forum/#!forum/digits-users
- For more information on getting started with DIGITS
  - Parallel forall blogs - http://devblogs.nvidia.com/parallelforall/easy-multi-gpu-deep-learning-digits-2/
  - Getting started guide - https://github.com/NVIDIA/DIGITS/blob/master/docs/GettingStarted.md

HANDS-ON LAB

1. Create an account at https://nvidia.qwiklab.com/
2. Go to “Getting Started with DIGITS” lab at
3. Start the lab and enjoy!

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

DEEP LEARNING LAB SERIES SCHEDULE

- All classes start at 9am PT

- 8/5 Class #2 - Getting Started with DIGITS interactive training system for image classification
- 8/12 Office Hours for Class #2

- 8/19 Class #3 - Getting Started with the Caffe Framework
- 8/26 Office Hours for Class #3

- 9/2 Class #4 - Getting Started with the Theano Framework
- 9/9 Office Hours for Class #4

- 9/16 Class #5 - Getting Started with the Torch Framework
- 9/23 Office Hours for Class #5

- More information available at developer.nvidia.com/deep-learning-courses