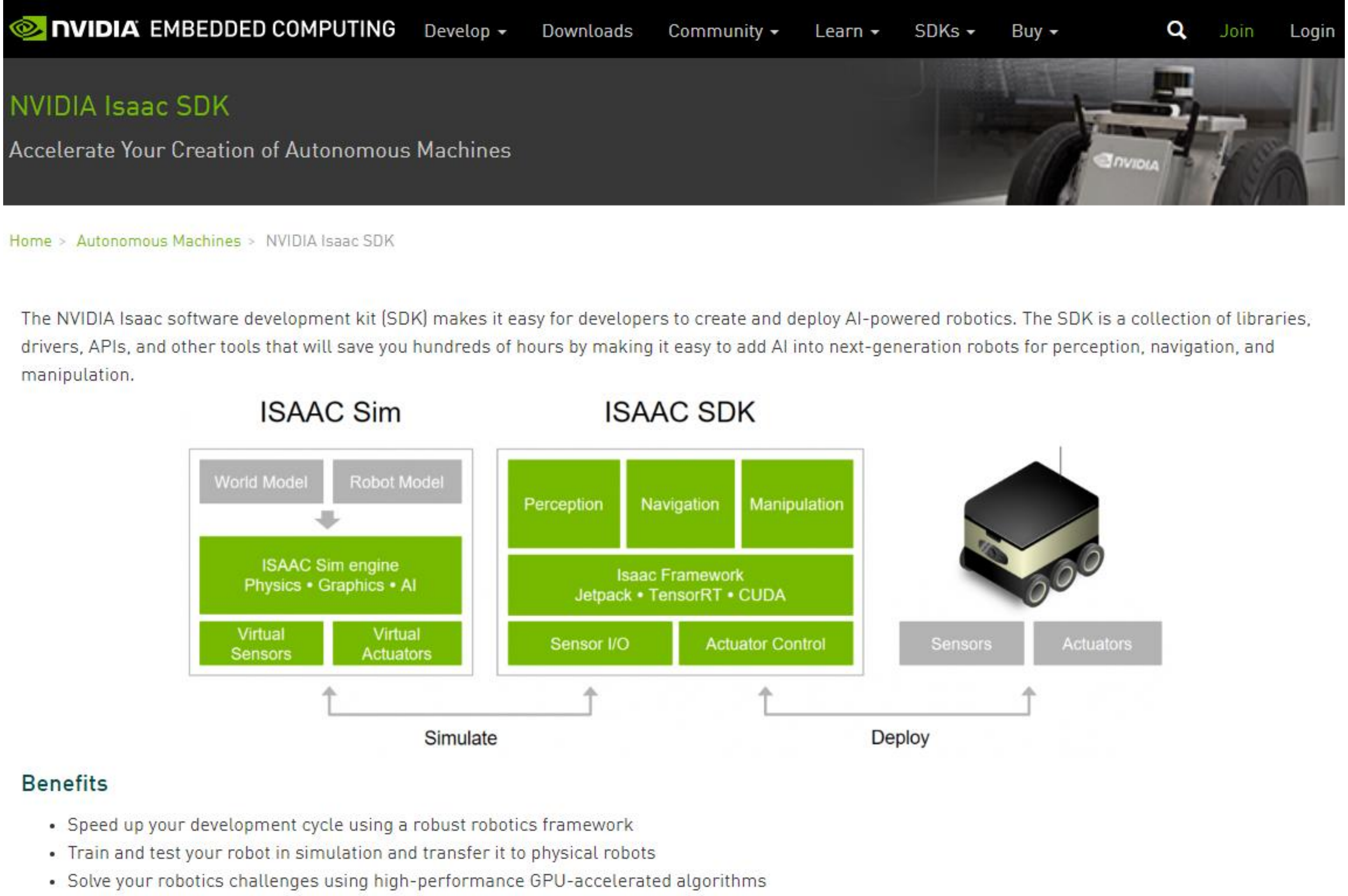
DEEP
LEARNING
INSTITUTE

不用写代码
使用NVIDIA DIGITS进行图像分类

侯宇涛
Developer marketing Director
Certified Instructor, NVIDIA Deep Learning Institute
NVIDIA China

ISAAC VIDEO

https://www.youtube.com/watch?v=PIDJdKwo2VM



NVIDIA Isaac SDK

Accelerate Your Creation of Autonomous Machines

The NVIDIA Isaac software development kit (SDK) makes it easy for developers to create and deploy AI-powered robotics. The SDK is a collection of libraries, drivers, APIs, and other tools that will save you hundreds of hours by making it easy to add AI into next-generation robots for perception, navigation, and manipulation.

Benefits

- Speed up your development cycle using a robust robotics framework
- Train and test your robot in simulation and transfer it to physical robots
- Solve your robotics challenges using high-performance GPU-accelerated algorithms

动手实验培训

通用基础 +专业应用领域

Caffe2

Microsoft Cognitive Toolkit

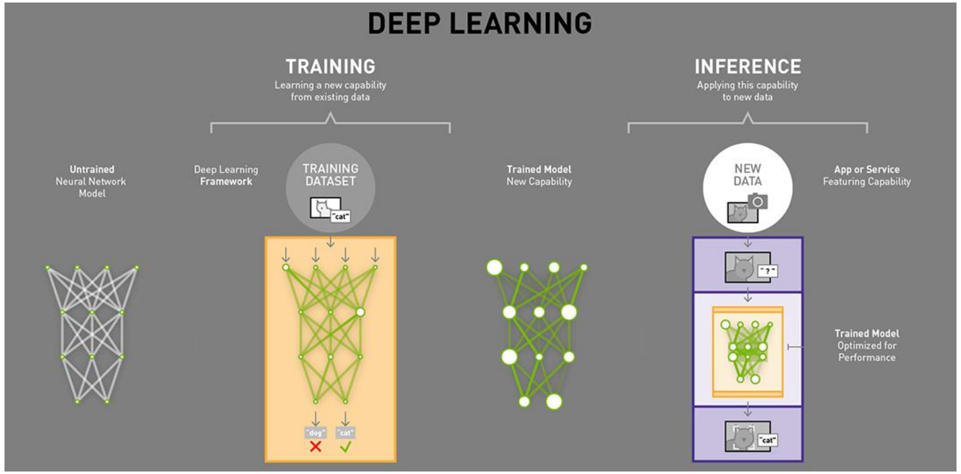
Microsoft

mxnet

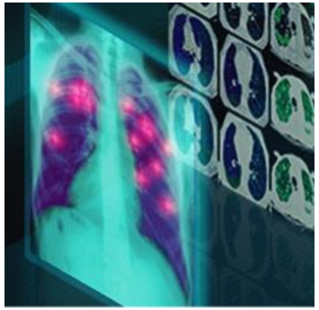
TensorFlow

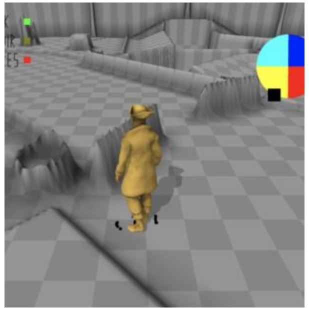
PYTORCH

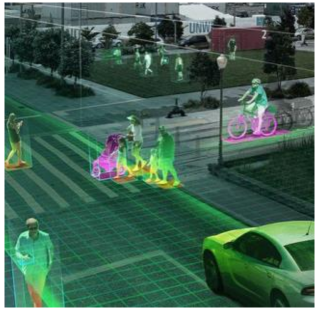
Fundamentals

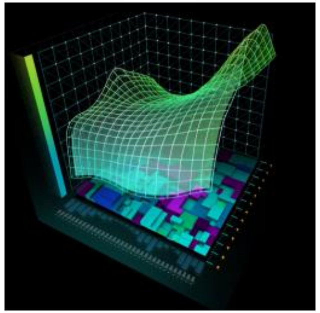

DEEP LEARNING
TRAINING
Learning a new capability from existing data
INFERENCE
Applying this capability to new data
Untrained Neural Network Model
Deep Learning Framework
TRAINING DATASET
"cat"
"dog"
"cat"
Trained Model New Capability
NEW DATA
"?"
"cat"
App or Service Featuring Capability
Trained Model Optimized for Performance

Autonomous Vehicles


Healthcare


Game Development and Digital Content

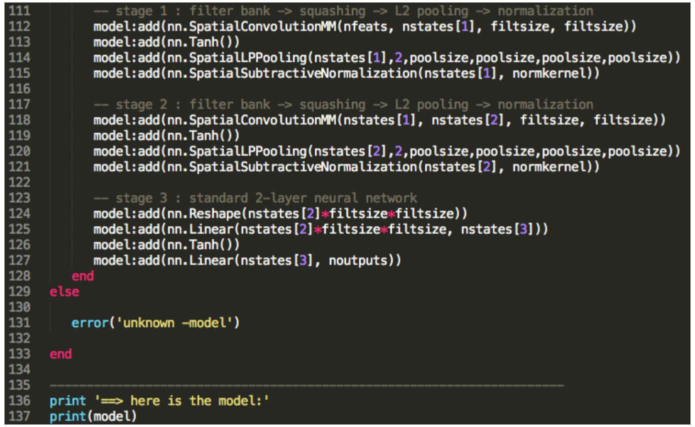

Intelligent Video Analytics


Finance

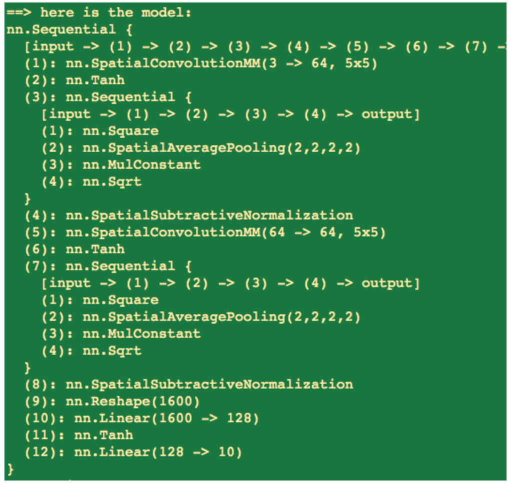
More content in a range of topics coming soon…

Traditional Deep Learning Workflow

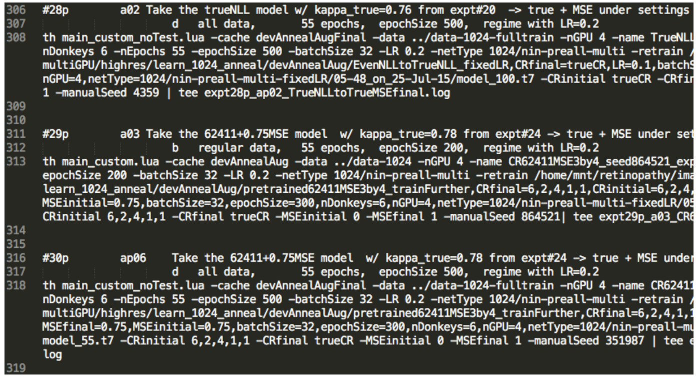
- DL frameworks, Caffe, etc. aimed at computer scientist not data scientist
- Juggle multiple files & windows
- Handcrafted visualizations
- Manual log file parsing
- Manual experiment logging
- Model editing in Lua IDE files



```
        -- stage 1 : filter bank -> squashing -> L2 pooling -> normalization
        model:add(nn.SpatialConvolutionMM(nfeats, nstates[1], filtsize, filtsize))
        model:add(nn.Tanh())
        model:add(nn.SpatialLPPooling(nstates[1],2,poolsize,poolsize,poolsize,poolsize))
        model:add(nn.SpatialSubtractiveNormalization(nstates[1], normkernel))

        -- stage 2 : filter bank -> squashing -> L2 pooling -> normalization
        model:add(nn.SpatialConvolutionMM(nstates[1], nstates[2], filtsize, filtsize))
        model:add(nn.Tanh())
        model:add(nn.SpatialLPPooling(nstates[2],2,poolsize,poolsize,poolsize,poolsize))
        model:add(nn.SpatialSubtractiveNormalization(nstates[2], normkernel))

        -- stage 3 : standard 2-layer neural network
        model:add(nn.Reshape(nstates[2]*filtsize*filtsize))
        model:add(nn.Linear(nstates[2]*filtsize*filtsize, nstates[3]))
        model:add(nn.Tanh())
        model:add(nn.Linear(nstates[3], noutputs))
    end
else

    error('unknown -model')

end

----------------------------------------------------------------------
print '==> here is the model:'
print(model)
```



```
==> here is the model:
nn.Sequential {
  [input -> (1) -> (2) -> (3) -> (4) -> (5) -> (6) -> (7) -
  (1): nn.SpatialConvolutionMM(3 -> 64, 5x5)
  (2): nn.Tanh
  (3): nn.Sequential {
    [input -> (1) -> (2) -> (3) -> (4) -> output]
    (1): nn.Square
    (2): nn.SpatialAveragePooling(2,2,2,2)
    (3): nn.MulConstant
    (4): nn.Sqrt
  }
  (4): nn.SpatialSubtractiveNormalization
  (5): nn.SpatialConvolutionMM(64 -> 64, 5x5)
  (6): nn.Tanh
  (7): nn.Sequential {
    [input -> (1) -> (2) -> (3) -> (4) -> output]
    (1): nn.Square
    (2): nn.SpatialAveragePooling(2,2,2,2)
    (3): nn.MulConstant
    (4): nn.Sqrt
  }
  (8): nn.SpatialSubtractiveNormalization
  (9): nn.Reshape(1600)
  (10): nn.Linear(1600 -> 128)
  (11): nn.Tanh
  (12): nn.Linear(128 -> 10)
}
```



```
#28p        a02 Take the trueNLL model w/ kappa_true=0.76 from expt#20  -> true + MSE under settings
                    d   all data,       55 epochs,  epochSize 500,  regime with LR=0.2
th main_custom_noTest.lua -cache devAnnealAugFinal -data ../data-1024-fulltrain -nGPU 4 -name TrueNLL
nDonkeys 6 -nEpochs 55 -epochSize 500 -batchSize 32 -LR 0.2 -netType 1024/nin-preall-multi -retrain /
multiGPU/highres/learn_1024_anneal/devAnnealAug/EvenNLLtoTrueNLL_fixedLR,CRfinal=trueCR,LR=0.1,batchS
nGPU=4,netType=1024/nin-preall-multi-fixedLR/05-48_on_25-Jul-15/model_100.t7 -CRinitial trueCR -CRfin
1 -manualSeed 4359 | tee expt28p_ap02_TrueNLLtoTrueMSEfinal.log


#29p        a03 Take the 62411+0.75MSE model  w/ kappa_true=0.78 from expt#24 -> true + MSE under set
                    b   regular data,   55 epochs,  epochSize 200,  regime with LR=0.2
th main_custom.lua -cache devAnnealAug -data ../data-1024 -nGPU 4 -name CR62411MSE3by4_seed864521_exp
epochSize 200 -batchSize 32 -LR 0.2 -netType 1024/nin-preall-multi -retrain /home/mnt/retinopathy/ima
learn_1024_anneal/devAnnealAug/pretrained62411MSE3by4_trainFurther,CRfinal=6,2,4,1,1,CRinitial=6,2,4,
MSEinitial=0.75,batchSize=32,epochSize=300,nDonkeys=6,nGPU=4,netType=1024/nin-preall-multi-fixedLR/05
CRinitial 6,2,4,1,1 -CRfinal trueCR -MSEinitial 0 -MSEfinal 1 -manualSeed 864521| tee expt29p_a03_CR6


#30p        ap06    Take the 62411+0.75MSE model  w/ kappa_true=0.78 from expt#24 -> true + MSE under
                    d   all data,       55 epochs,  epochSize 500,  regime with LR=0.2
th main_custom_noTest.lua -cache devAnnealAugFinal -data ../data-1024-fulltrain -nGPU 4 -name CR62411
nDonkeys 6 -nEpochs 55 -epochSize 500 -batchSize 32 -LR 0.2 -netType 1024/nin-preall-multi -retrain /
multiGPU/highres/learn_1024_anneal/devAnnealAug/pretrained62411MSE3by4_trainFurther,CRfinal=6,2,4,1,1
MSEfinal=0.75,MSEinitial=0.75,batchSize=32,epochSize=300,nDonkeys=6,nGPU=4,netType=1024/nin-preall-mu
model_55.t7 -CRinitial 6,2,4,1,1 -CRfinal trueCR -MSEinitial 0 -MSEfinal 1 -manualSeed 351987 | tee e
log

```

课程介绍：

级别：初级 | 预备知识：无
行业：所有 | Frameworks: Caffe

此实验室会向您展示如何通过在 Caffe 框架上的 NVIDIA DIGITS 和MNIST 手写数据集，在深度学习工作流程中利用深度神经网络 (DNN)， 尤其是卷积神经网络 (CNN) 解决真实图像分类问题，您会学到：

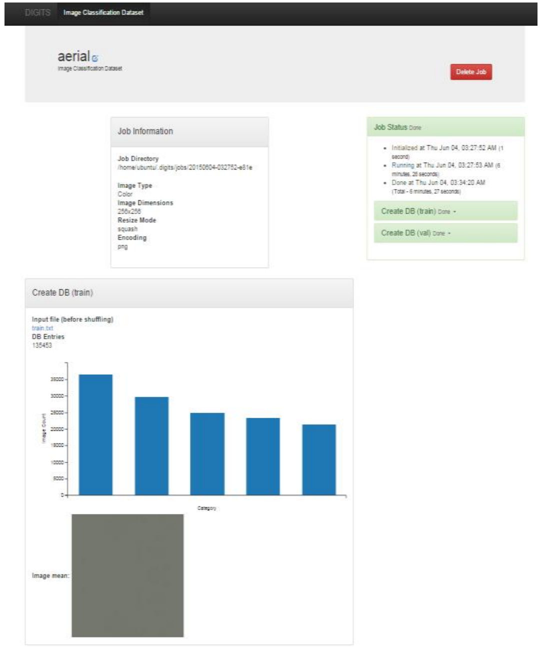
1. 构建运行在GPU上的深度神经网络
2. 管理数据准备、模型定义、模型训练和问题排查过程
3. 使用验证数据来测试和尝试不同策略来提升模型性能

完成此实验室后，您将能够使用 NVIDIA DIGITS 来构建、训练、评估和提升您的图像分类应用程序中 CNN 的准确性。

NVIDIA DIGITS

交互式深度学习GPU训练系统

Process Data
加载数据集



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

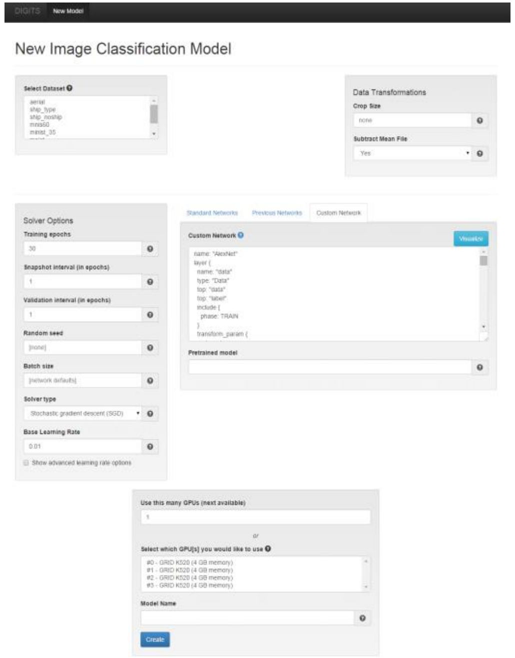
Create DB (val) Done

Create DB (train)

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

Image mean:

Configure DNN
配置神经网络



DIGITS New Model

New Image Classification Model

Select Dataset
aerial
ship_type
ship_noship
mnis60
mnist_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

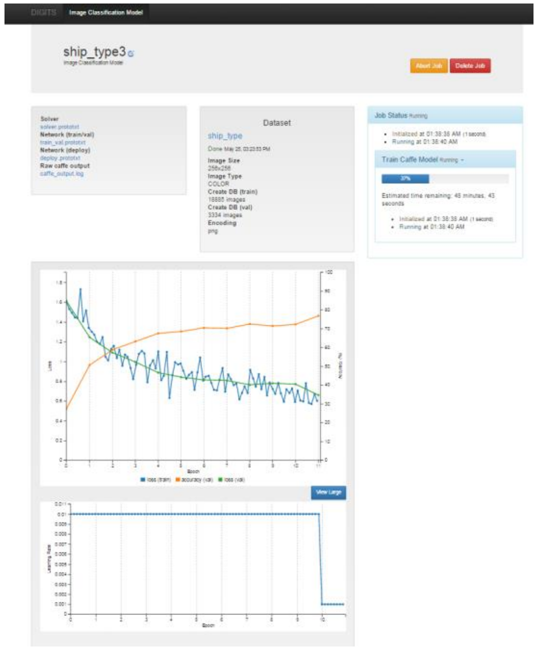
or

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
监控训练过程



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 28, 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

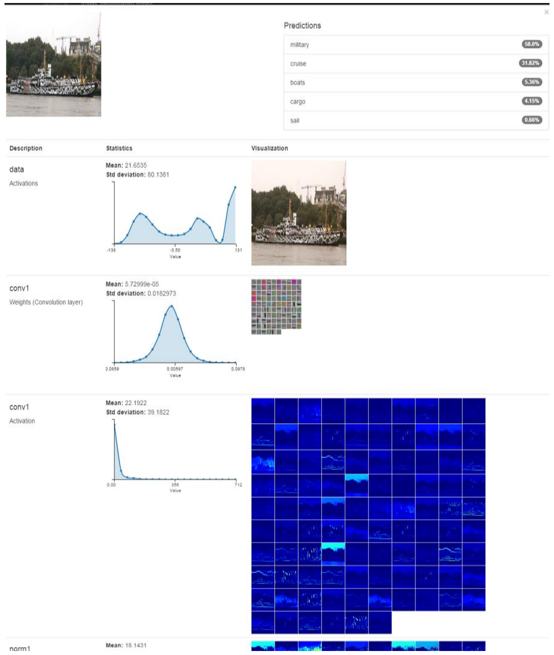
Train Caffe Model Running

30%

Estimated time remaining: 48 minutes, 43 seconds

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

Visualization
可视化校验



Predictions

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

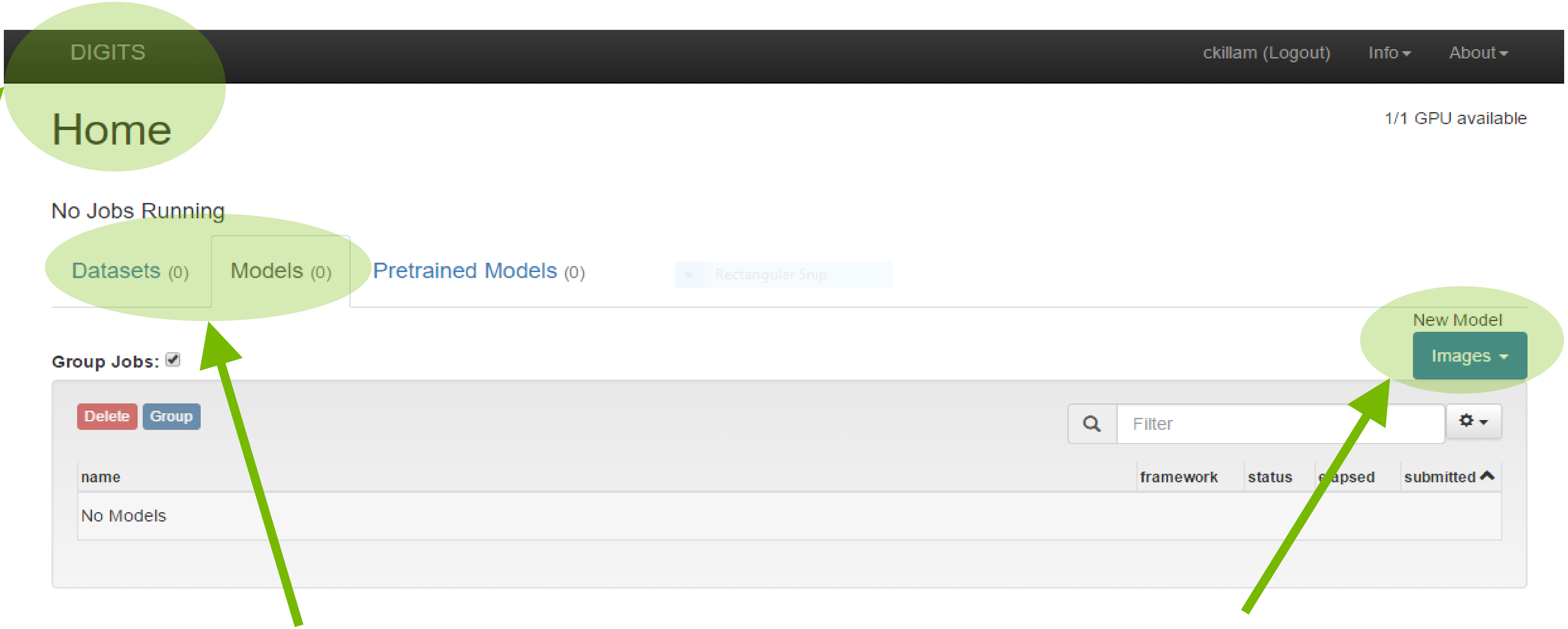
Description	Statistics	Visualization
data Activations	**Mean:** 21.6535 **Std deviation:** 80.1381	
conv1 Weights (Convolution layer)	**Mean:** 5.72999e-05 **Std deviation:** 0.0182973	
conv1 Activation	**Mean:** 22.1922 **Std deviation:** 39.1822	
norm1	**Mean:** 18.1431	

NVIDIA’S DIGITS

交互式深度学习GPU训练系统

- 简化通用的深度学习任务如：
  - 管理数据
  - 在多GPU系统上设计并训练神经网络
  - 使用高级可视化界面，监控实时性能
- 完全的交互式界面，使得数据科学家可以专注在设计及训练网络，节约编程及调试代码的时间
- 开源

DIGITS - HOME

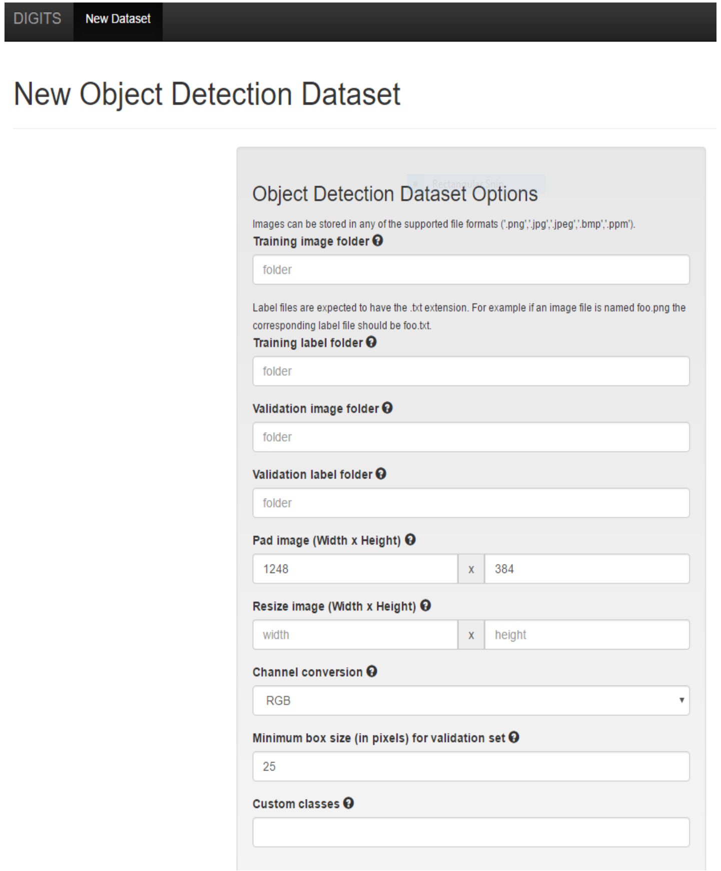


Clicking DIGITS will bring you to this Home screen

Click here to see a list of existing datasets or models

Clicking here will present different options for model and dataset creation

DIGITS - DATASET



New Object Detection Dataset

Object Detection Dataset Options

Images can be stored in any of the supported file formats ('.png','.jpg','.jpeg','.bmp','.ppm').

Training image folder
folder

Label files are expected to have the .txt extension. For example if an image file is named foo.png the corresponding label file should be foo.txt.

Training label folder
folder

Validation image folder
folder

Validation label folder
folder

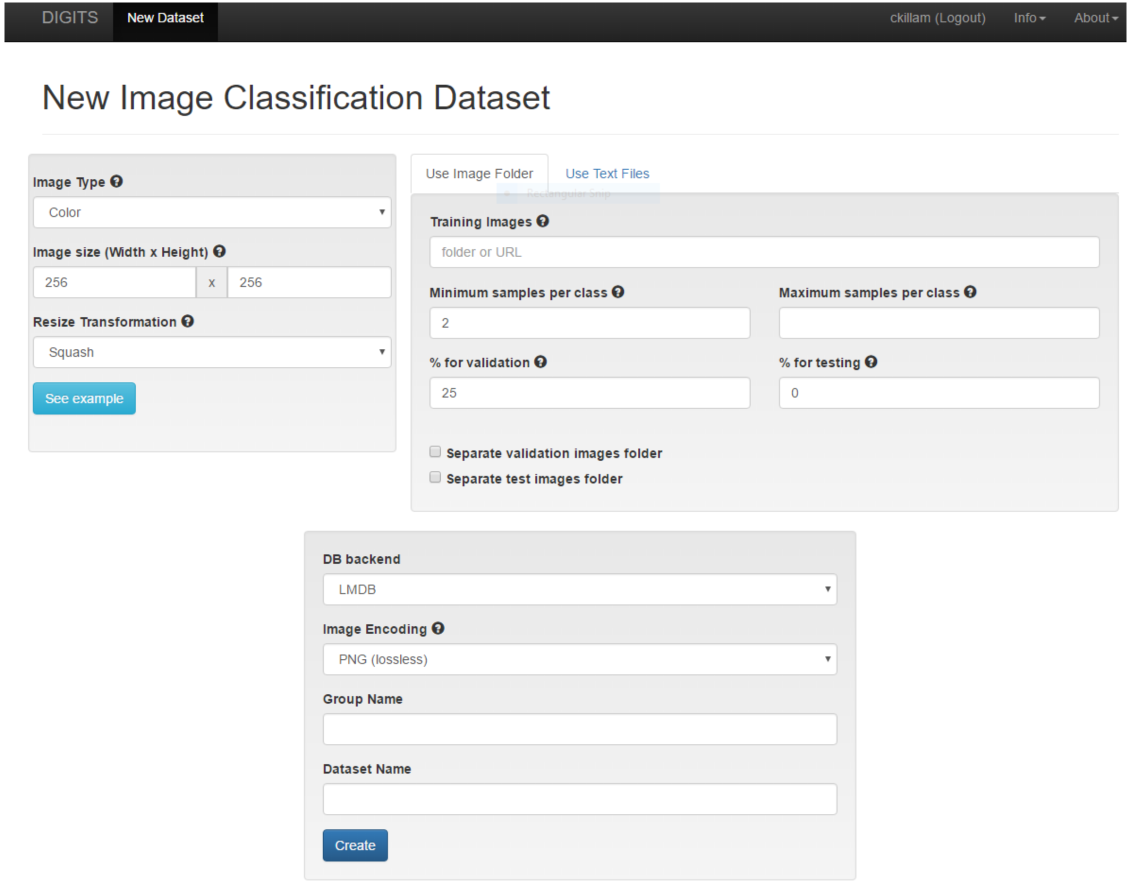
Pad image (Width x Height)
1248 x 384

Resize image (Width x Height)
width x height

Channel conversion
RGB

Minimum box size (in pixels) for validation set
25

Custom classes



New Image Classification Dataset

Image Type
Color

Image size (Width x Height)
256 x 256

Resize Transformation
Squash

Training Images
folder or URL

Minimum samples per class
2

Maximum samples per class

% for validation
25

% for testing
0

☐ Separate validation images folder
☐ Separate test images folder

DB backend
LMDB

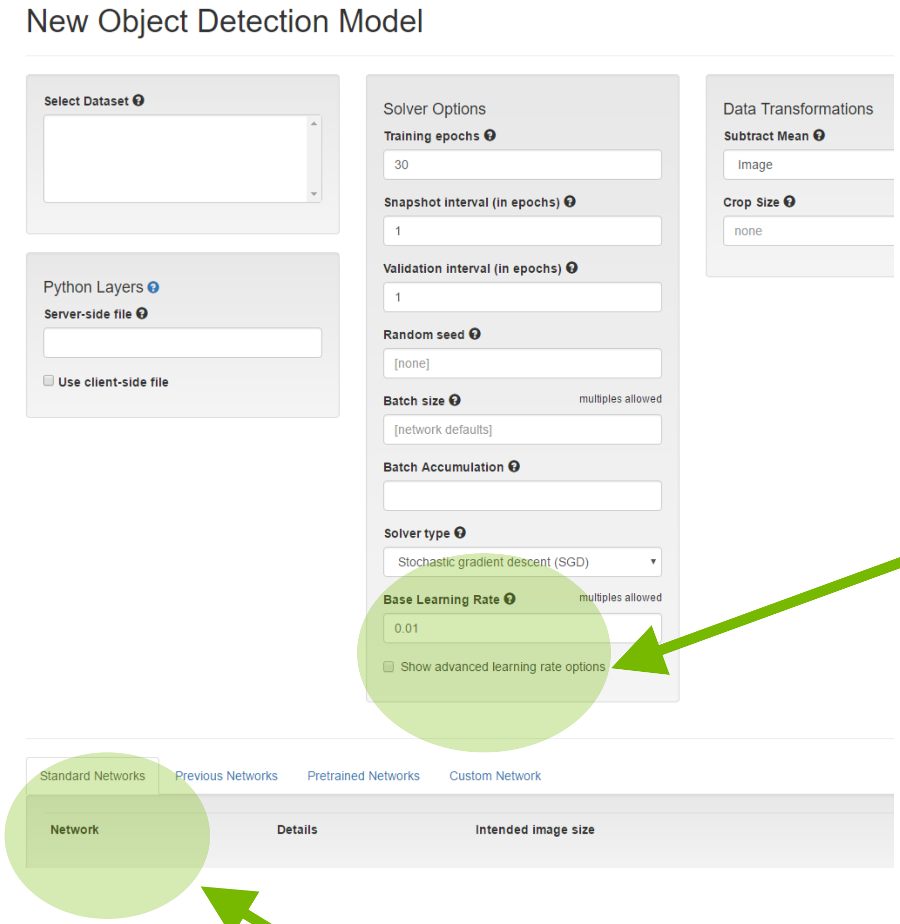
Image Encoding
PNG (lossless)

Group Name

Dataset Name

Different options will be presented based upon the task

DIGITS - MODEL



New Object Detection Model

Select Dataset

Python Layers

Server-side file

☐ Use client-side file

Solver Options

Training epochs: 30

Snapshot interval (in epochs): 1

Validation interval (in epochs): 1

Random seed: [none]

Batch size (multiples allowed): [network defaults]

Batch Accumulation

Solver type: Stochastic gradient descent (SGD)

Base Learning Rate (multiples allowed): 0.01

☐ Show advanced learning rate options

Data Transformations

Subtract Mean: Image

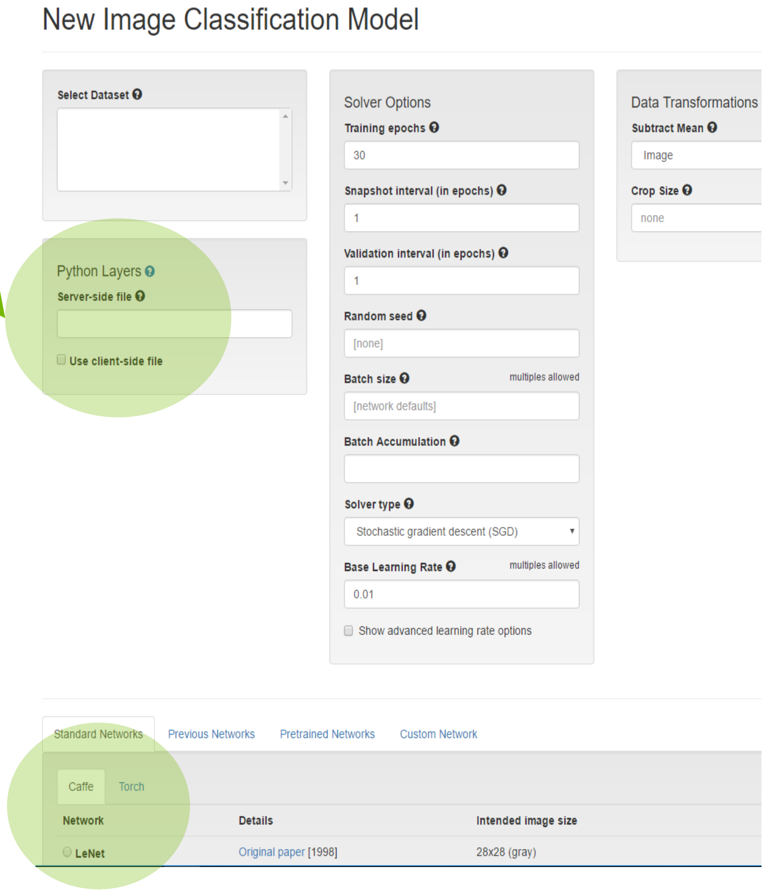
Crop Size: none

Standard Networks | Previous Networks | Pretrained Networks | Custom Network

Network	Details	Intended image size

Define custom layers with Python

Can anneal the learning rate



New Image Classification Model

Select Dataset

Python Layers

Server-side file

☐ Use client-side file

Solver Options

Training epochs: 30

Snapshot interval (in epochs): 1

Validation interval (in epochs): 1

Random seed: [none]

Batch size (multiples allowed): [network defaults]

Batch Accumulation

Solver type: Stochastic gradient descent (SGD)

Base Learning Rate (multiples allowed): 0.01

☐ Show advanced learning rate options

Data Transformations

Subtract Mean: Image

Crop Size: none

Standard Networks | Previous Networks | Pretrained Networks | Custom Network

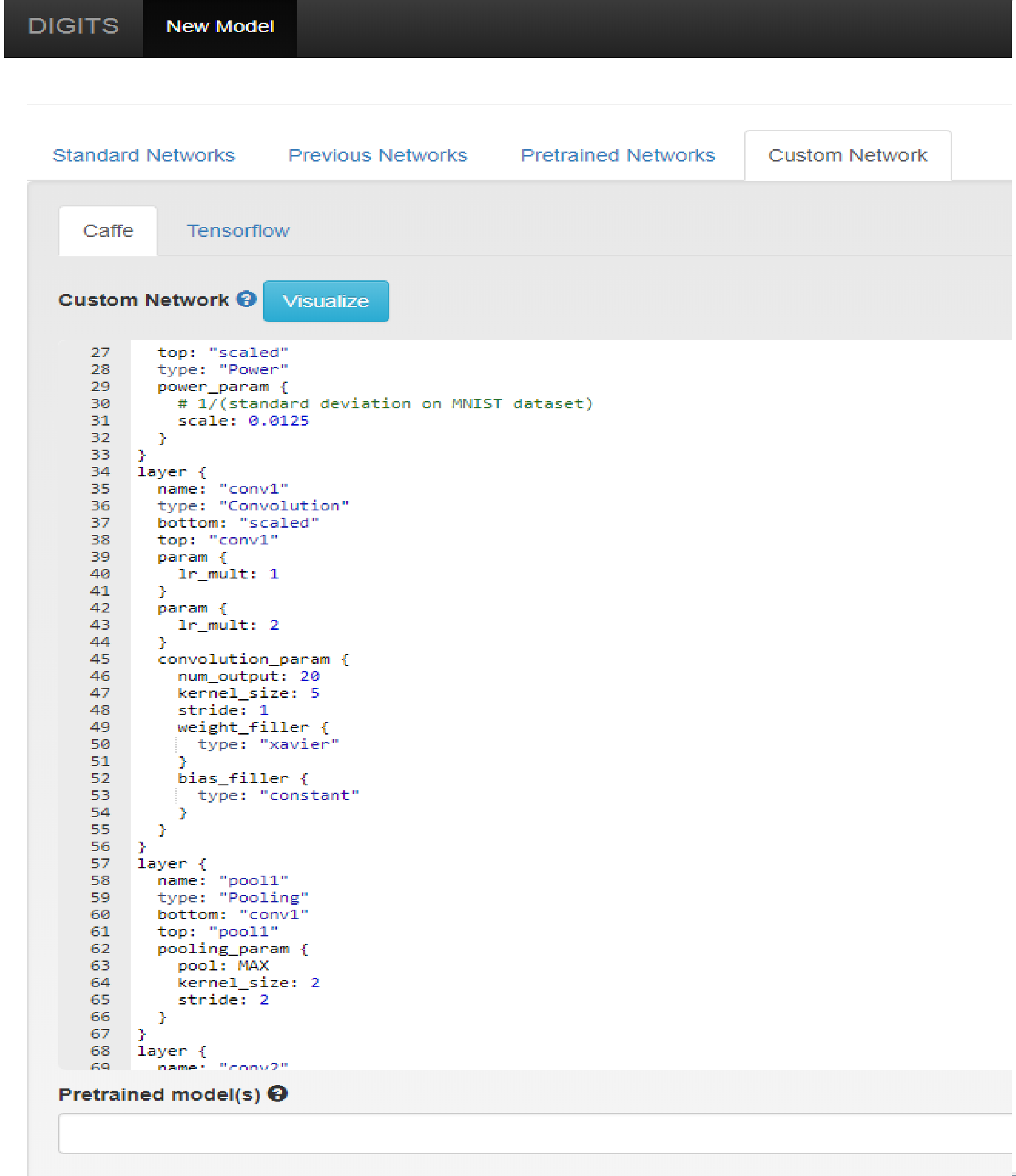
Caffe | Torch

Network	Details	Intended image size
○ LeNet	Original paper [1998]	28x28 (gray)

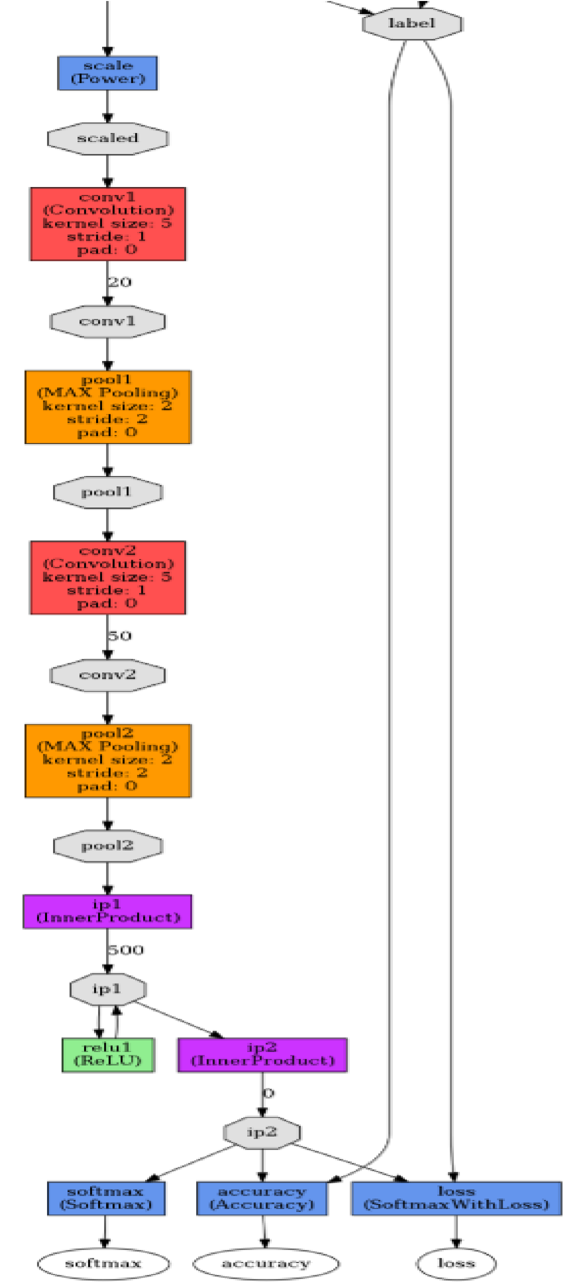
Differences may exist between model tasks

DIGITS

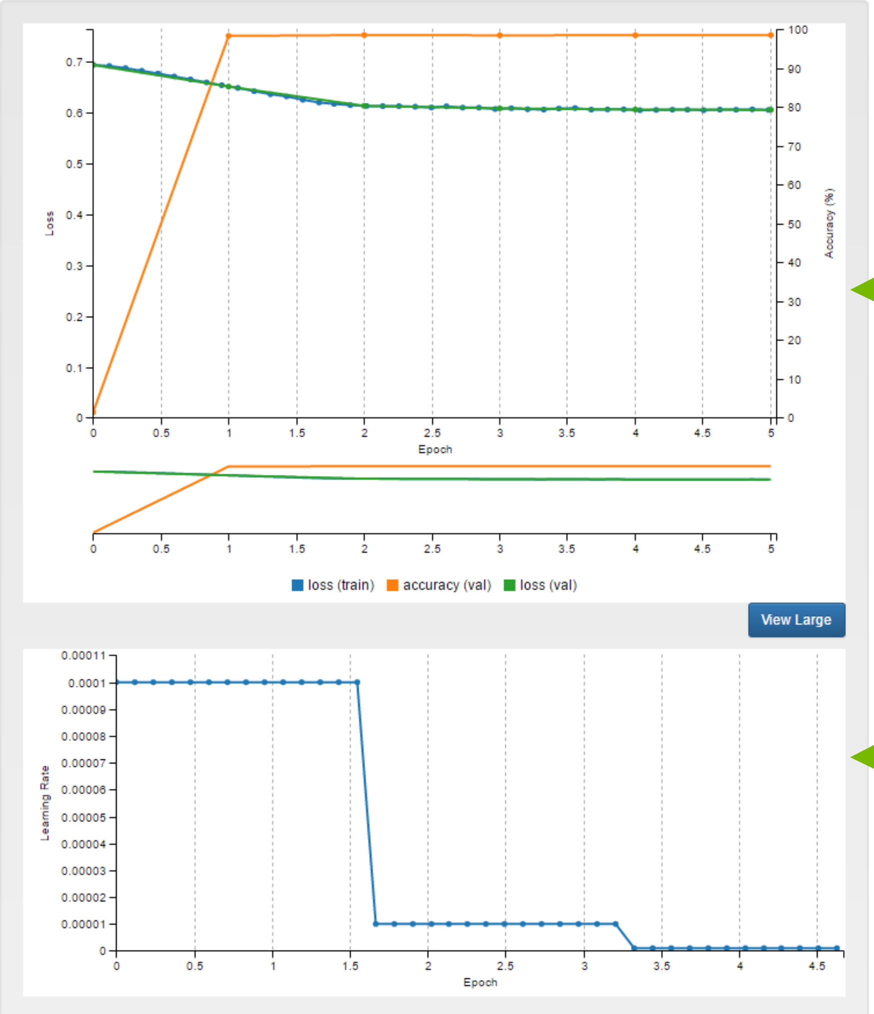
- MODEL



```
  top: "scaled"
  type: "Power"
  power_param {
    # 1/(standard deviation on MNIST dataset)
    scale: 0.0125
  }
}
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "scaled"
  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"
    }
  }
}
layer {
  name: "pool1"
  type: "Pooling"
  bottom: "conv1"
  top: "pool1"
  pooling_param {
    pool: MAX
    kernel_size: 2
    stride: 2
  }
}
layer {
  name: "conv2"
```



DIGITS - TRAINING

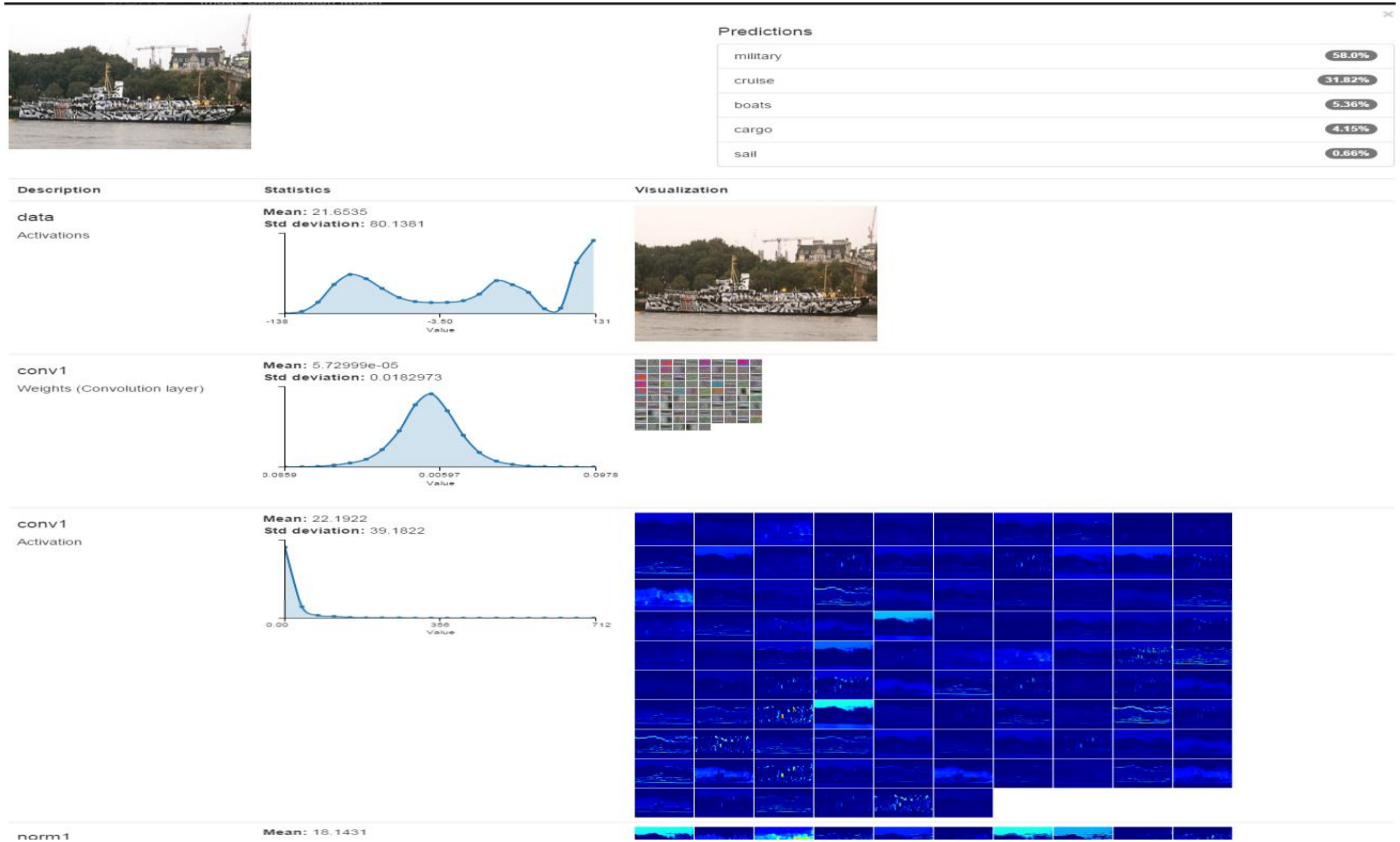


Loss function and accuracy during training

Annealed learning rate

NVIDIA DIGITS

Visualization

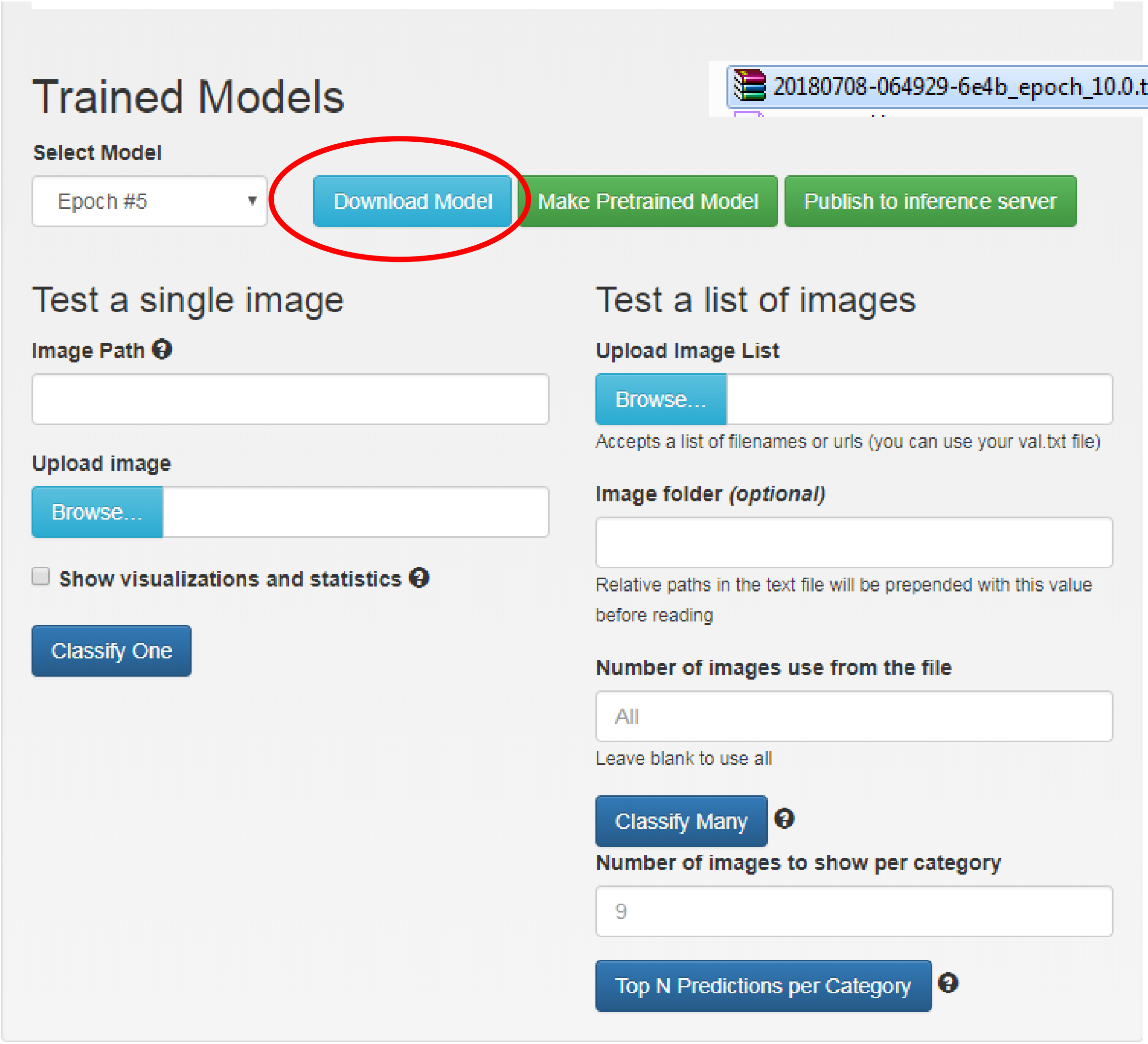


Predictions

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



Trained Models

Select Model

Epoch #5 ▾

Test a single image

Image Path

Upload image

□ Show visualizations and statistics

Test a list of images

Upload Image List

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

Image folder *(optional)*

Relative paths in the text file will be prepended with this value before reading

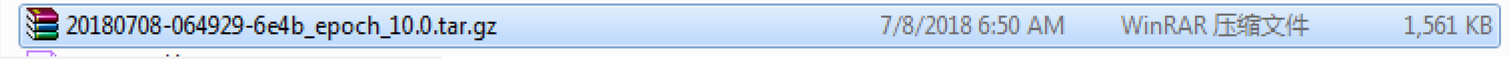
Number of images use from the file

All

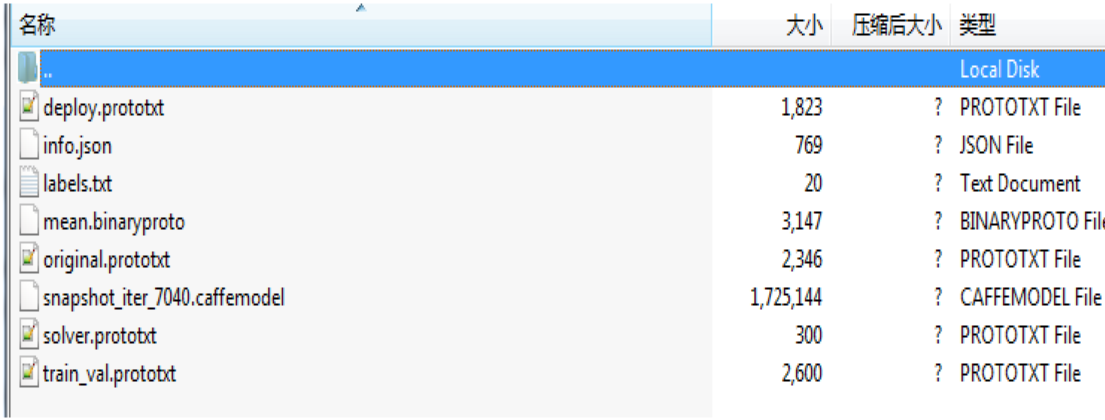
Leave blank to use all

Number of images to show per category

9



20180708-064929-6e4b_epoch_10.0.tar.gz   7/8/2018 6:50 AM   WinRAR 压缩文件   1,561 KB



名称	大小	压缩后大小	类型
..			Local Disk
deploy.prototxt	1,823	?	PROTOTXT File
info.json	769	?	JSON File
labels.txt	20	?	Text Document
mean.binaryproto	3,147	?	BINARYPROTO File
original.prototxt	2,346	?	PROTOTXT File
snapshot_iter_7040.caffemodel	1,725,144	?	CAFFEMODEL File
solver.prototxt	300	?	PROTOTXT File
train_val.prototxt	2,600	?	PROTOTXT File

DIGITS DEEP LEARNING Workflows

IMAGE CLASSIFICATION

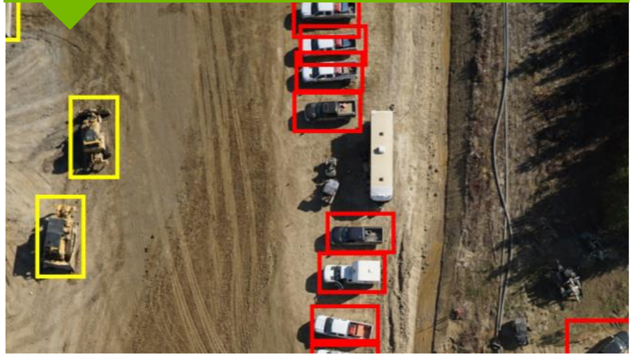
98% Dog

2% Cat

Classify images into classes or categories

Object of interest could be anywhere in the image

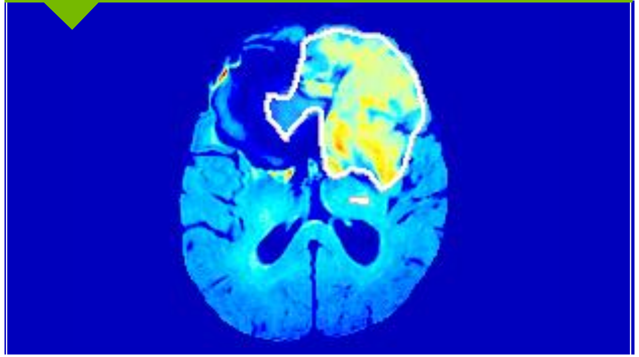
OBJECT DETECTION



Find instances of objects in an image

Objects are identified with bounding boxes

IMAGE SEGMENTATION



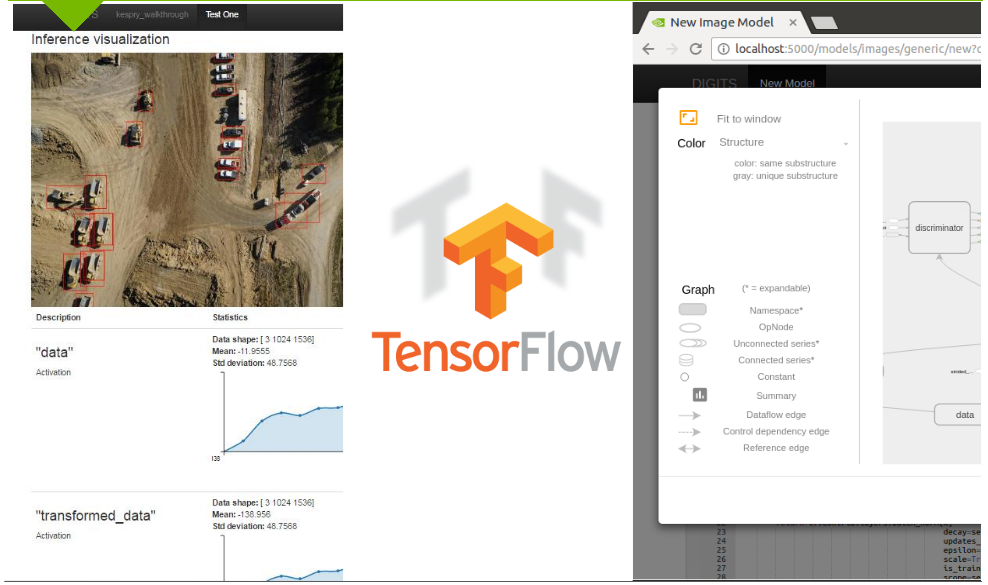
Partition image into multiple regions

Regions are classified at the pixel level

developer.nvidia.com/digits

NVIDIA digits 6

TENSORFLOW SUPPORT

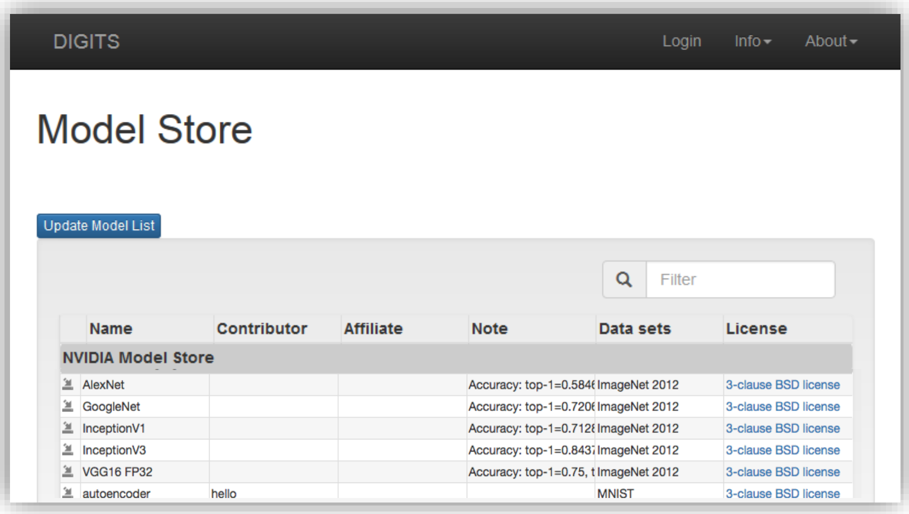


Inference visualization

Description	Statistics
"data" Activation	Data shape: [3 1024 1536] Mean: -11.9555 Std deviation: 48.7568
"transformed_data" Activation	Data shape: [3 1024 1536] Mean: -138.956 Std deviation: 48.7568

Train TensorFlow Models Interactively with DIGITS

NEW PRE-TRAINED MODELS



Model Store

	Name	Contributor	Affiliate	Note	Data sets	License
NVIDIA Model Store						
	AlexNet			Accuracy: top-1=0.584(	ImageNet 2012	3-clause BSD license
	GoogleNet			Accuracy: top-1=0.720(	ImageNet 2012	3-clause BSD license
	InceptionV1			Accuracy: top-1=0.712(	ImageNet 2012	3-clause BSD license
	InceptionV3			Accuracy: top-1=0.843	ImageNet 2012	3-clause BSD license
	VGG16 FP32			Accuracy: top-1=0.75, t	ImageNet 2012	3-clause BSD license
	autoencoder	hello			MNIST	3-clause BSD license

Image Classification: VGG-16, ResNet50
Object Detection: DetectNet

DIGITS 6 is now available now as a free download to members of NVIDIA Developer Program

developer.nvidia.com/digits

TUTORIAL教程:

HANDWRITTEN DIGIT RECOGNITION
手写体数字识别模型训练及优化

HANDWRITTEN DIGIT RECOGNITION

HELLO WORLD of machine learning?

MNIST data set of handwritten digits from Yann Lecun’s website

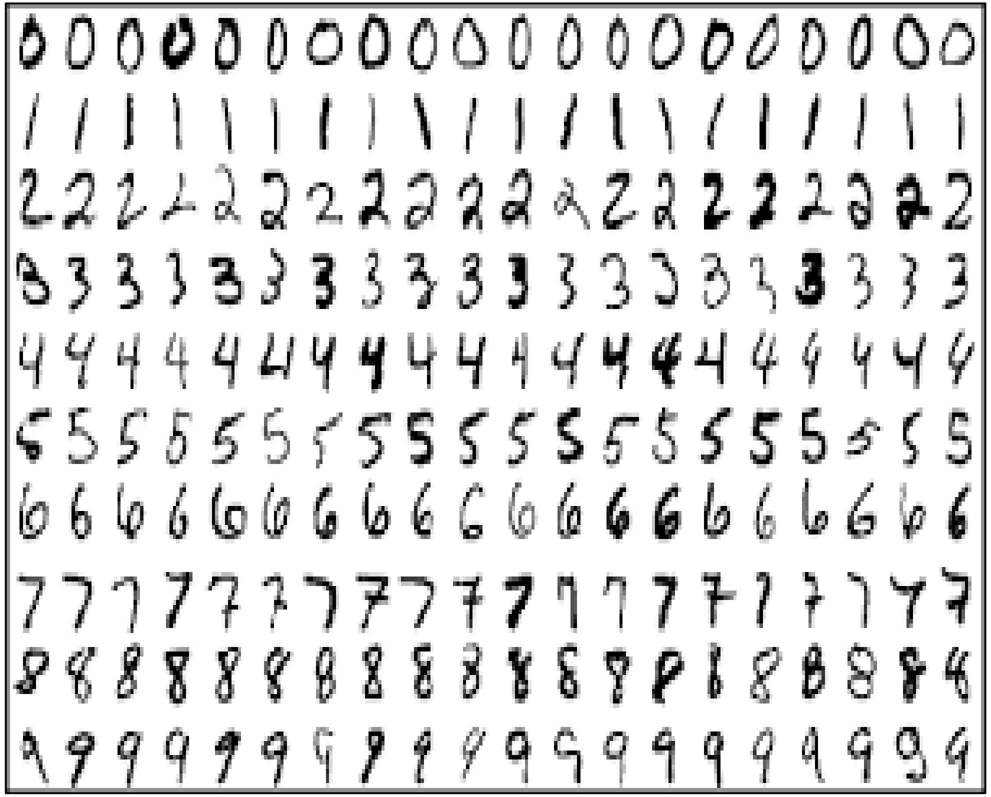
All images are 28x28 grayscale

Pixel values from 0 to 255

60k training examples, 10k test examples

Input vector of size 784

Output value is integer from 0-9



SMALL DATASET

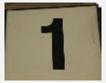
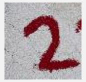
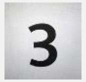
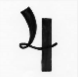
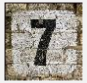
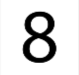
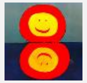
6000 x images

- Dataset
  - Training Images: /home/ubuntu/data/train_small
- Model
  - “MNIST small”

FIRST RESULTS

Small dataset (10 epochs)

- 96% of accuracy achieved
- Training is done within one minute

	SMALL DATASET
	1 : 99.90 %
	2 : 69.03 %
	8 : 71.37 %
	8 : 85.07 %
	0 : 99.00 %
	8 : 99.69 %
	8 : 54.75 %

FULL DATASET

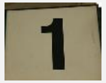
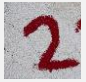
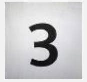
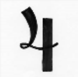
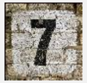
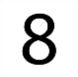
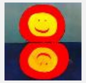
6x larger dataset

- Dataset
  - Training Images: /home/ubuntu/data/train_full
  - Dataset Name: MNIST full
- Model
  - Clone “MNIST small”.
  - Give a new name “MNIST full” to push the create button

SECOND RESULTS

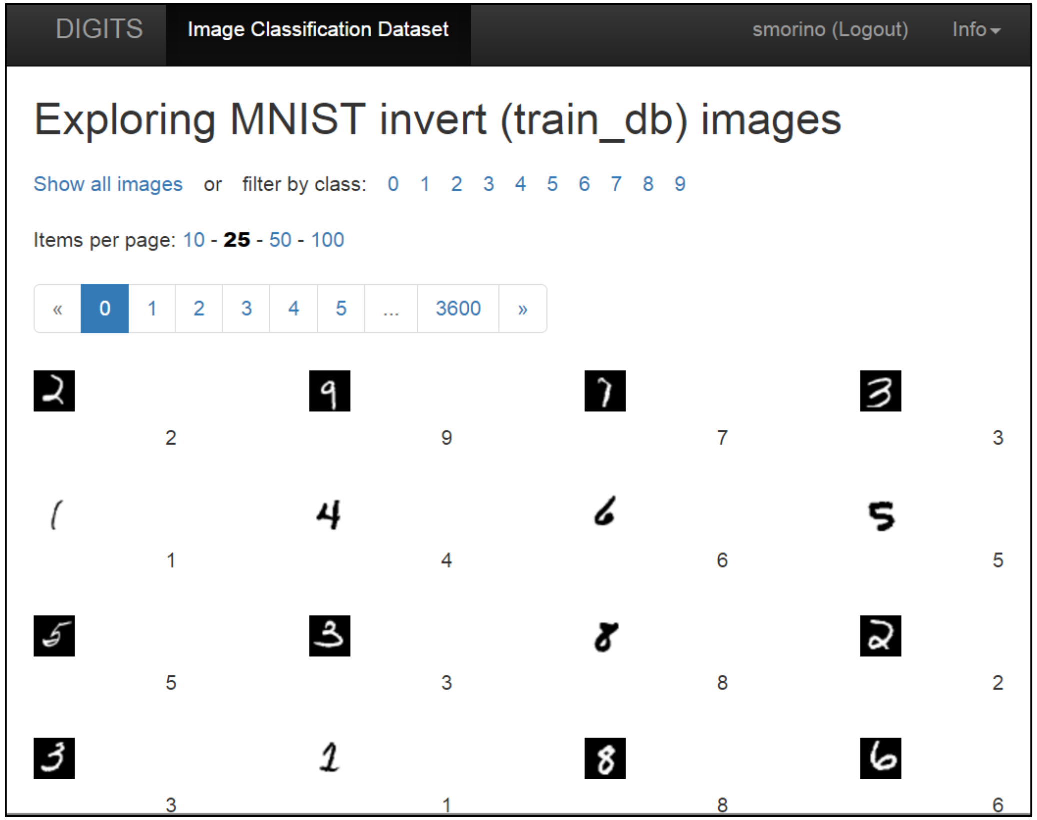
Full dataset (10 epochs)

- 99% of accuracy achieved
- No improvements in recognizing real-world images

	SMALL DATASET	FULL DATASET
	1 : 99.90 %	0 : 93.11 %
	2 : 69.03 %	2 : 87.23 %
	8 : 71.37 %	8 : 71.60 %
	8 : 85.07 %	8 : 79.72 %
	0 : 99.00 %	0 : 95.82 %
	8 : 99.69 %	8 : 100.0 %
	8 : 54.75 %	2 : 70.57 %

DATA AUGMENTATION

Adding Inverted Images



Exploring MNIST invert (train_db) images

Show all images or filter by class: 0 1 2 3 4 5 6 7 8 9

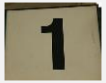
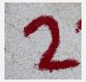
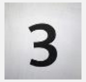
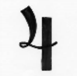
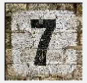
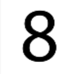
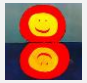
Items per page: 10 - **25** - 50 - 100

« 0 1 2 3 4 5 ... 3600 »

- Pixel(Inverted) = 255 – Pixel(original)
- White letter with black background
  - Black letter with white background
- Training Images: /home/ubuntu/data/train_invert
- Dataset Name: MNIST invert

DATA AUGMENTATION

Adding inverted images (10 epochs)

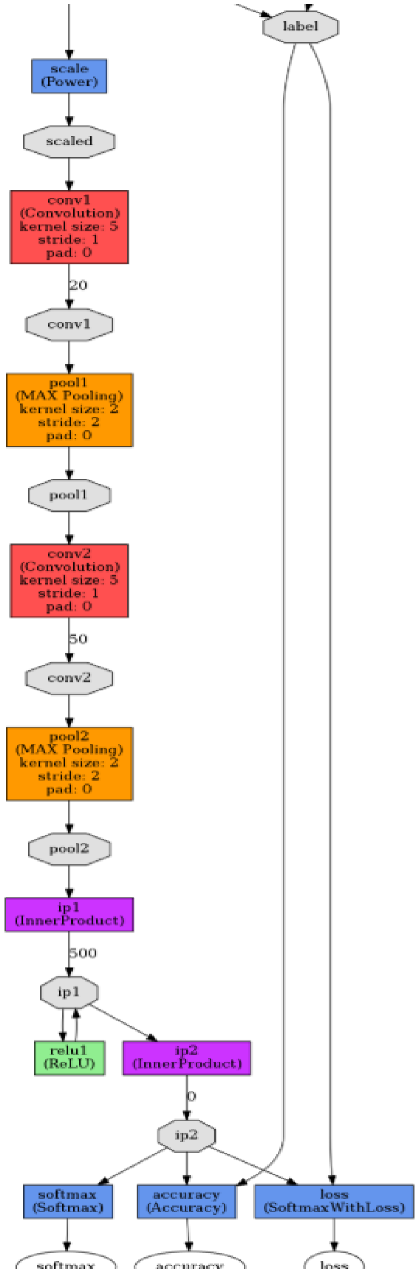
	SMALL DATASET	FULL DATASET	+INVERTED
	1 : 99.90 %	0 : 93.11 %	1 : 90.84 %
	2 : 69.03 %	2 : 87.23 %	2 : 89.44 %
	8 : 71.37 %	8 : 71.60 %	3 : 100.0 %
	8 : 85.07 %	8 : 79.72 %	4 : 100.0 %
	0 : 99.00 %	0 : 95.82 %	7 : 82.84 %
	8 : 99.69 %	8 : 100.0 %	8 : 100.0 %
	8 : 54.75 %	2 : 70.57 %	2 : 96.27 %

MODIFY THE NETWORK

Adding filters and ReLU layer

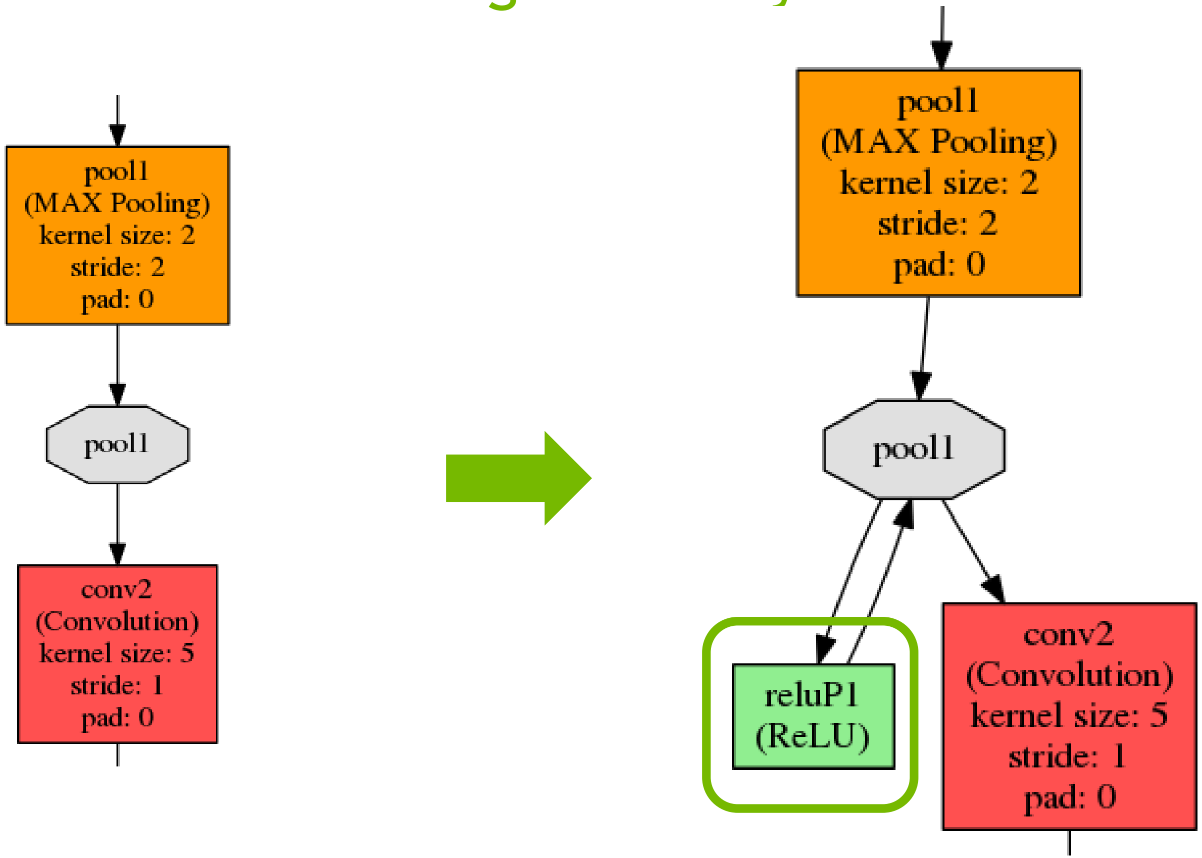
```
layer {
        name: "pool1“
        type: "Pooling“
        …
}

layer {
        name: "reluP1"
        type: "ReLU"
        bottom: "pool1"
        top: "pool1"
}

layer {
        name: "reluP1“
```

```
layer {
  name: "conv1"
  type: "Convolution"
        ...
        convolution_param {
        num_output: 75
        ...
layer {
        name: "conv2"
        type: "Convolution"
        ...
        convolution_param {
        num_output: 100
        ...
```



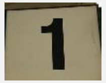
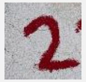
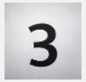
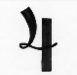
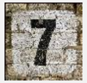
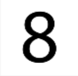
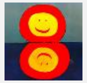
MODIFY THE NETWORK

Adding ReLU Layer

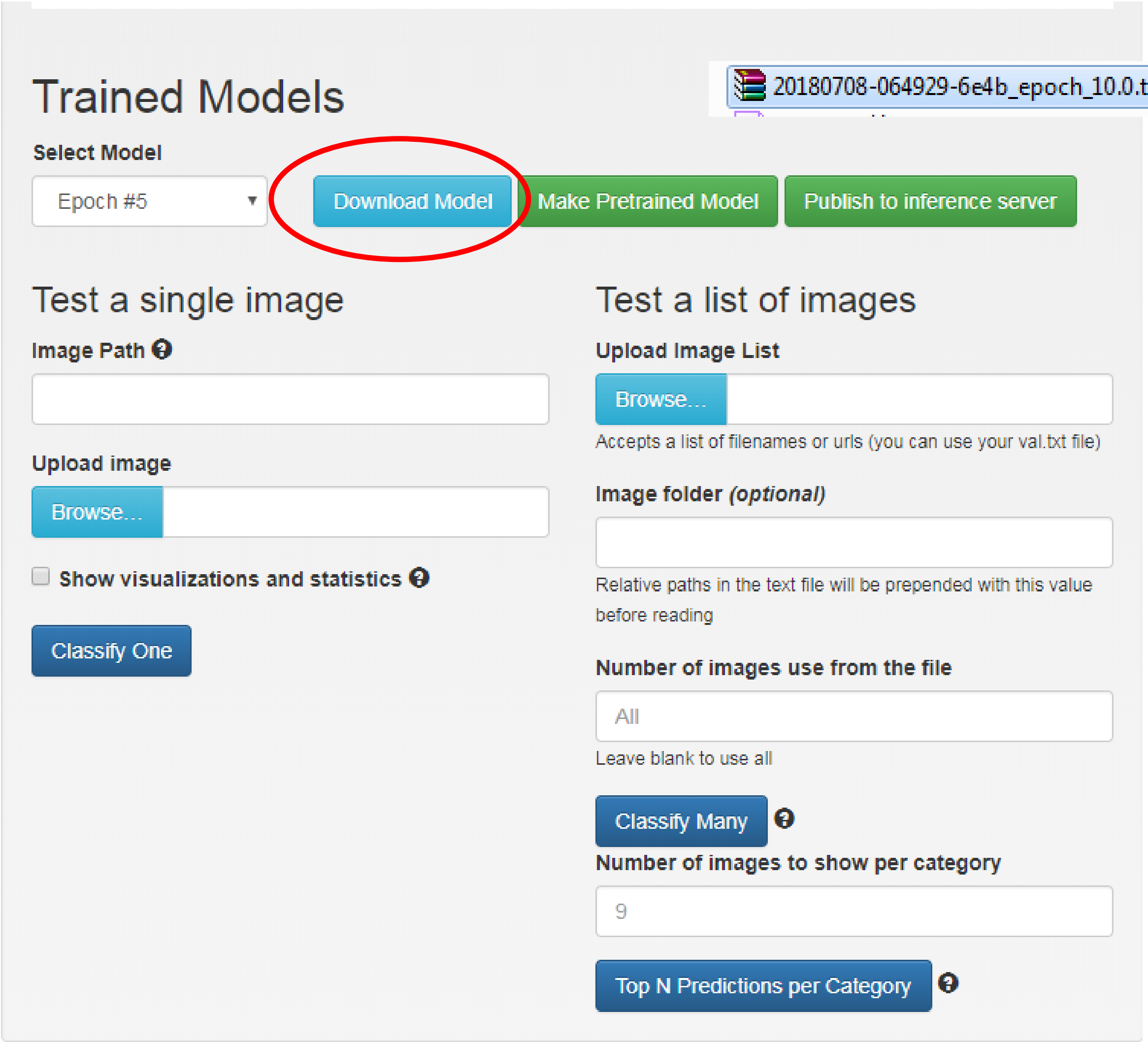


MODIFIED NETWORK

Adding filters and ReLU layer (10 epochs)

	SMALL DATASET	FULL DATASET	+INVERTED	ADDING LAYER
	1 : 99.90 %	0 : 93.11 %	1 : 90.84 %	1 : 59.18 %
	2 : 69.03 %	2 : 87.23 %	2 : 89.44 %	2 : 93.39 %
	8 : 71.37 %	8 : 71.60 %	3 : 100.0 %	3 : 100.0 %
	8 : 85.07 %	8 : 79.72 %	4 : 100.0 %	4 : 100.0 %
	0 : 99.00 %	0 : 95.82 %	7 : 82.84 %	2 : 62.52 %
	8 : 99.69 %	8 : 100.0 %	8 : 100.0 %	8 : 100.0 %
	8 : 54.75 %	2 : 70.57 %	2 : 96.27 %	8 : 70.83 %

NVIDIA DIGITS



Trained Models

Select Model

Epoch #5 ▾

Test a single image

Image Path

Upload image

□ **Show visualizations and statistics**

Test a list of images

Upload Image List

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

Image folder *(optional)*

Relative paths in the text file will be prepended with this value before reading

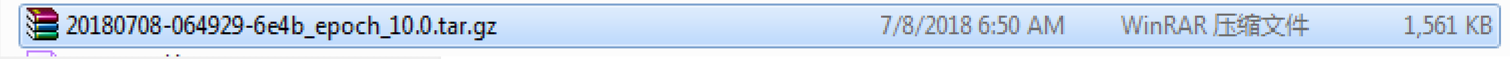
Number of images use from the file

All

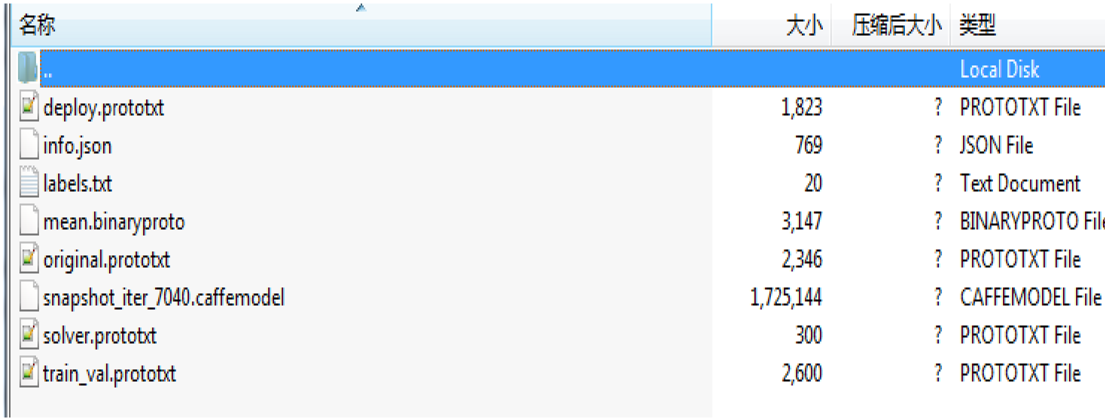
Leave blank to use all

Number of images to show per category

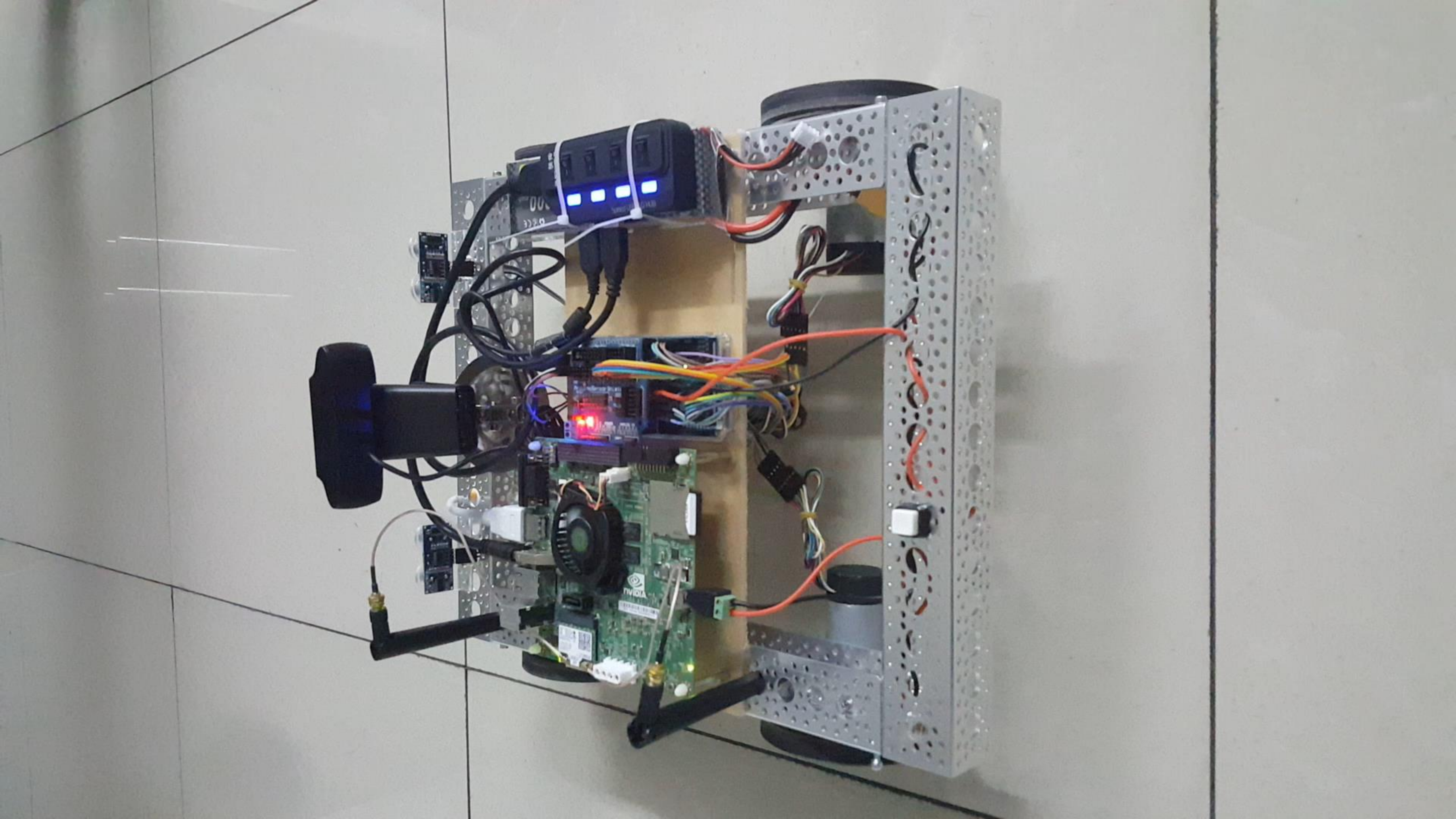
9



20180708-064929-6e4b_epoch_10.0.tar.gz 7/8/2018 6:50 AM WinRAR 压缩文件 1,561 KB



名称	大小	压缩后大小	类型
..			Local Disk
deploy.prototxt	1,823	?	PROTOTXT File
info.json	769	?	JSON File
labels.txt	20	?	Text Document
mean.binaryproto	3,147	?	BINARYPROTO Fil
original.prototxt	2,346	?	PROTOTXT File
snapshot_iter_7040.caffemodel	1,725,144	?	CAFFEMODEL File
solver.prototxt	300	?	PROTOTXT File
train_val.prototxt	2,600	?	PROTOTXT File


NVIDIA

How to get DIGITS

Two methods to install DIGITS

Simple way:

- OS – Ubuntu14.04
- Download link : https://developer.nvidia.com/digits

Others (from source code)：

- OSX，Windows (not tested)
- Download NVIDIA-Caffe：https://github.com/NVIDIA/caffe
- Download Digits：https://github.com/NVIDIA/DIGITS

Recommended HW/SW environment:

- GPU Compute Capability > 3.0 (Kepler and later)，cuDNN v5
- OS – Ubuntu14.04

NVIDIA 深度学习学院（DLI）

深度学习和加速计算培训
面向开发者、数据科学家和研究人员

- 真 · 实战培训，云端完全配置的 GPU 实验环境
- 全球同步，与 TensorFlow facebook 等机构共创课程
- 系统化培训，提升解决行业实际问题的能力
- NVIDIA “开发者认证”证书，构建职场竞争力

DLI 课程和开发者资源

“NVIDIA 开发者社区”

在线免费实训课
在 GPU 云上训练图像分类模型

不用写代码
使用NVIDIA DIGITS**进行图像分类**

7月26日 20:00-22:00
扫码进群