GPU TECHNOLOGY CONFERENCE

April 4-7, 2016 | Silicon Valley

# L6136 - DEEP LEARNING ON GPUS: FROM LARGE SCALE TRAINING TO EMBEDDED DEPLOYMENT

Julie Bernauer, 4/5/2016

PRESENTED BY

NVIDIA.

# AGENDA

0. Introduction
1. Connecting to the Amazon cloud
2. Train an image classification model using Digits
3. Caffe on the TX1: the fp16 whitepaper
4. Deploy the image classification model on TX1
5. Fun with the webcam

# INTRODUCTION

# ACCELERATING INSIGHTS

*“Now You Can Build Google’s $1M Artificial Brain on the Cheap”*

WIRED

## GOOGLE DATACENTER

1,000 CPU Servers
2,000 CPUs • 16,000 cores

**600 kWatts**
**$5,000,000**

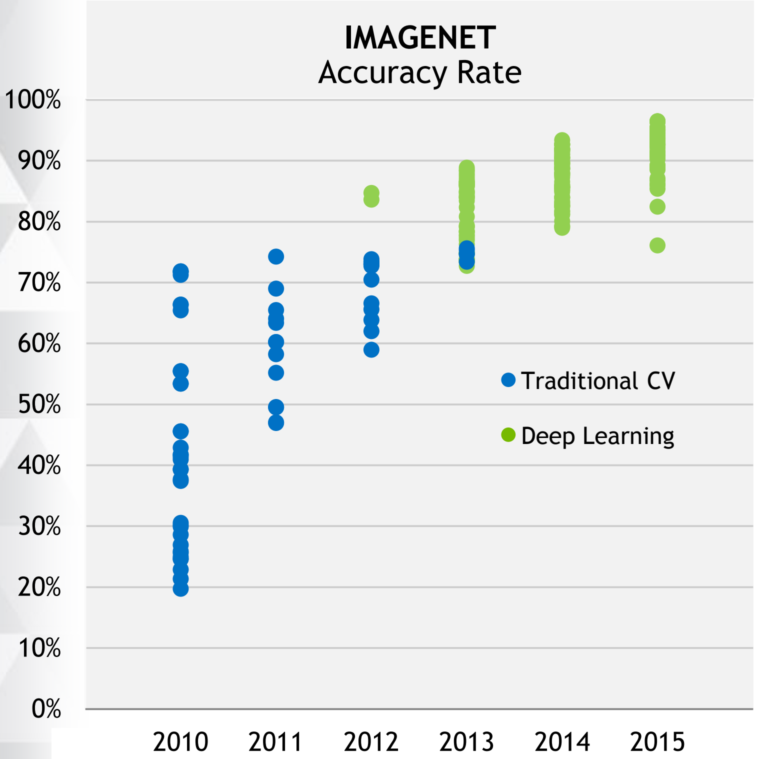
## STANFORD AI LAB

3 GPU-Accelerated Servers
12 GPUs • 18,432 cores

**4 kWatts**
**$33,000**

*Deep learning with COTS HPC systems*, A. Coates, B. Huval, T. Wang, D. Wu, A. Ng, B. Catanzaro ICML 2013

# MODERN AI

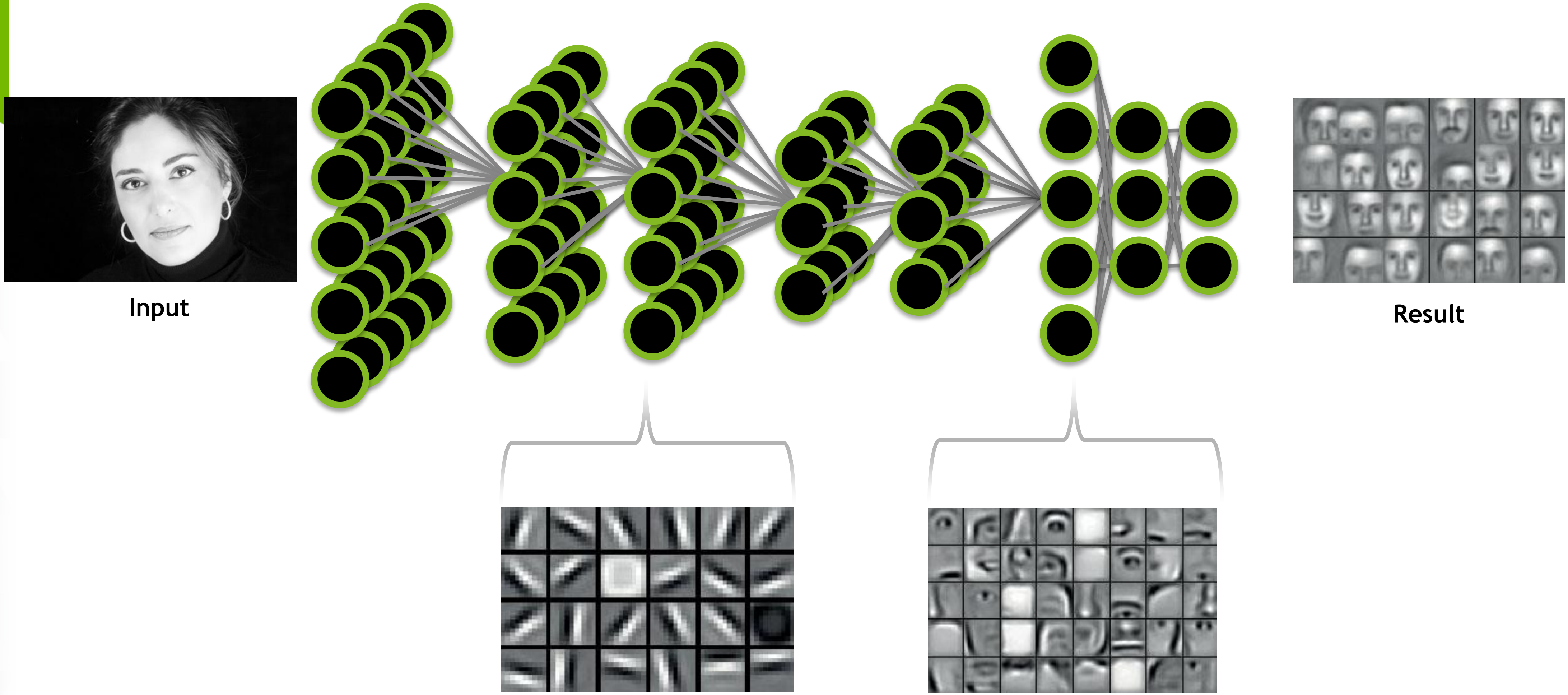


2012: GOOGLE BRAIN

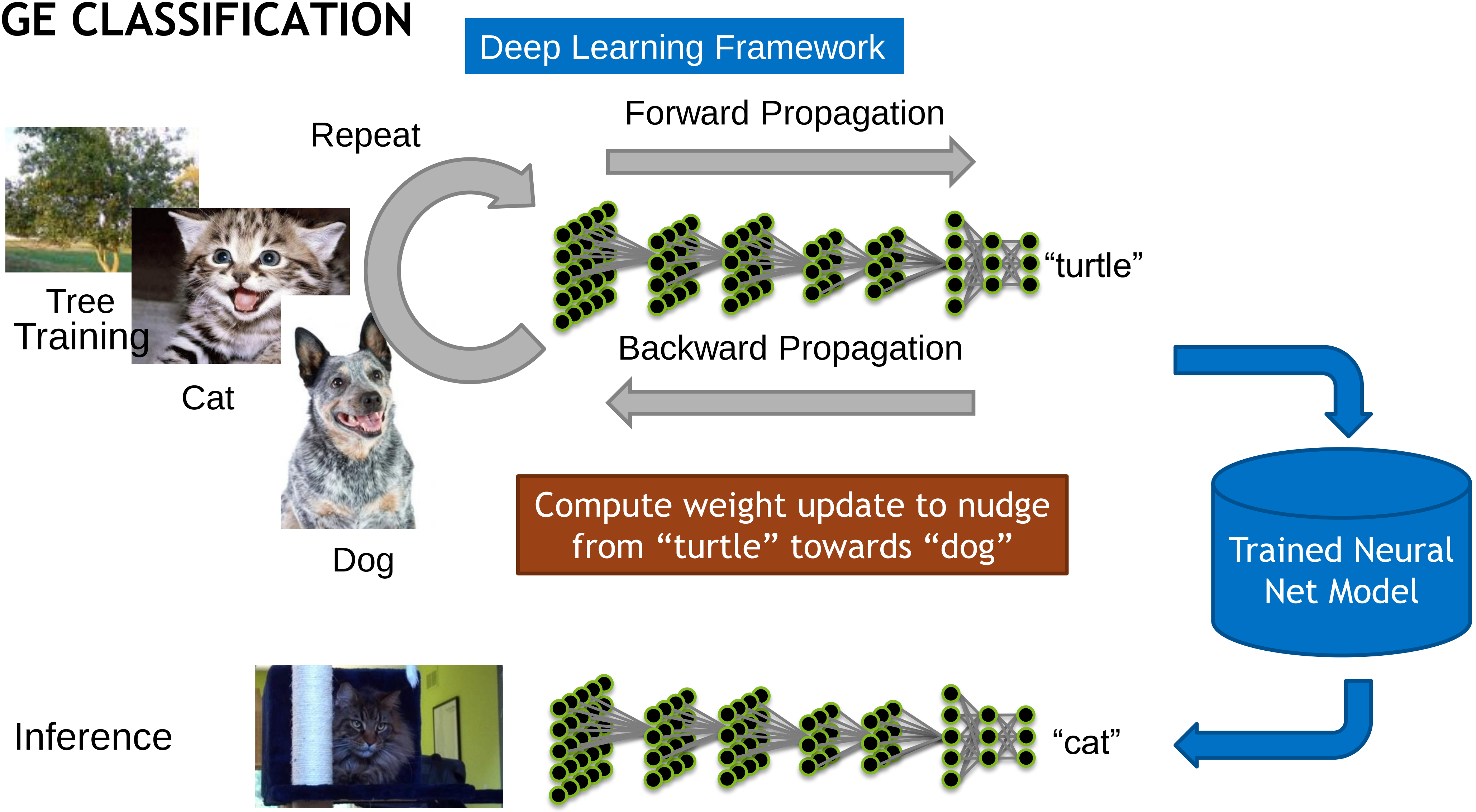
nature
THE INTERNATIONAL WEEKLY JOURNAL OF SCIENCE
At last — a computer program that can beat a champion Go player PAGE 484
ALL SYSTEMS GO

2016: AlphaGO

# WHAT IS DEEP LEARNING?



# IMAGE CLASSIFICATION



Deep Learning Framework

Training

Tree

Cat

Dog

Repeat

Forward Propagation

"turtle"

Backward Propagation

Compute weight update to nudge from "turtle" towards "dog"

Trained Neural Net Model

Inference

"cat"

# NVIDIA DEEP LEARNING SDK

## High Performance GPU-Acceleration for Deep Learning

IMAGENET

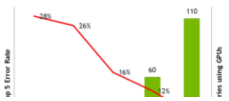
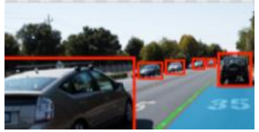


Image Classification



Object Detection

COMPUTER VISION

Voice Recognition

Language Translation

SPEECH AND AUDIO

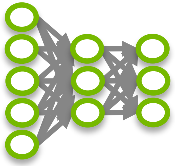
Recommendation Engines

Sentiment Analysis

NATURAL LANGUAGE PROCESSING

Caffe Chainer DL4J Deeplearning4j Mocha.jl julia KERAS Microsoft CNTK MatConvNet mxnet MINERVA OpenDeep Purine Pylearn2 TensorFlow torch theano

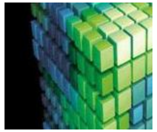
DEEP LEARNING FRAMEWORKS



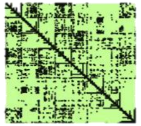
cuDNN

DEEP LEARNING

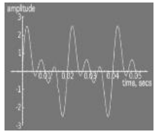
cuBLAS



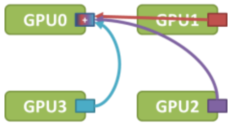
cuSPARSE



cuFFT



MATH LIBRARIES

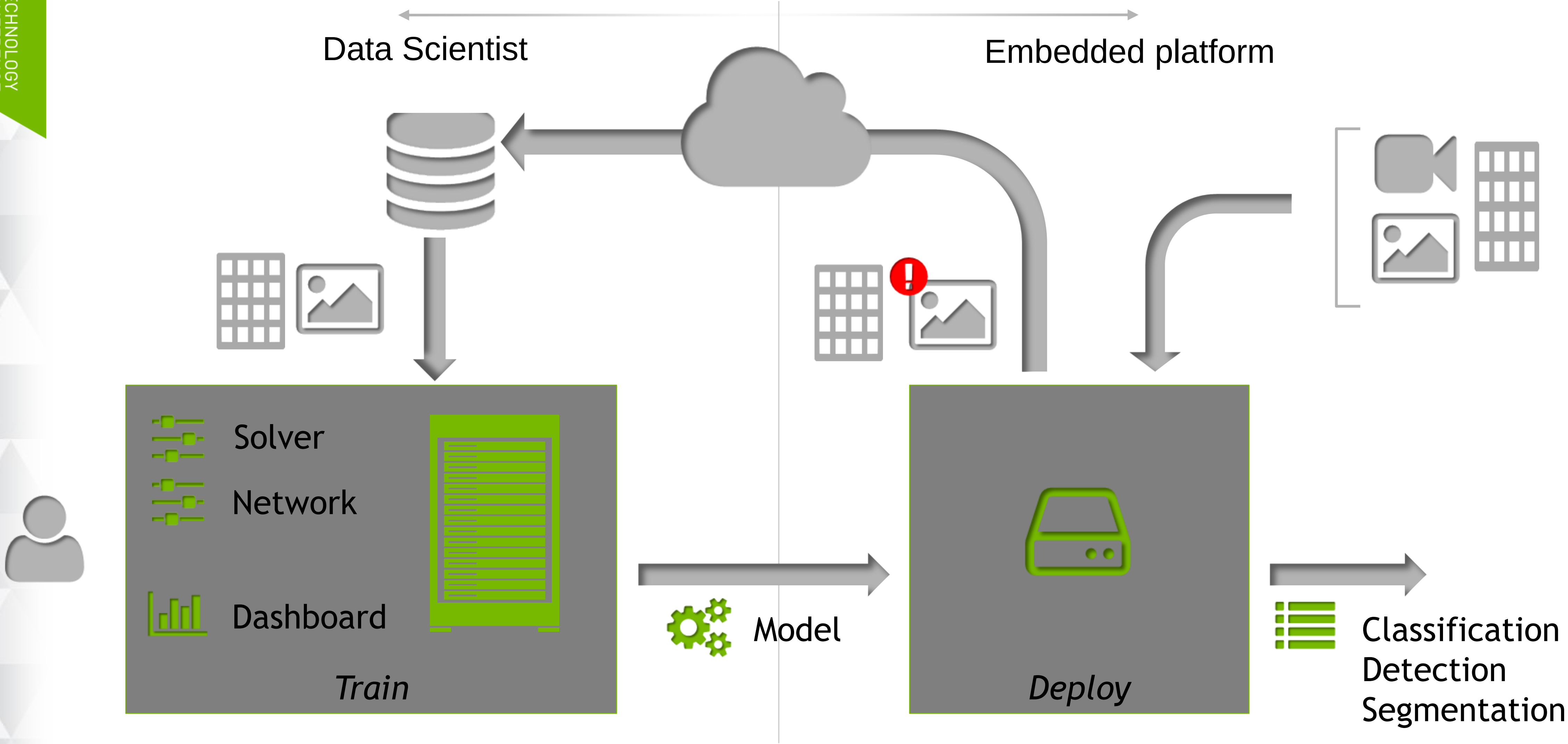


NCCL

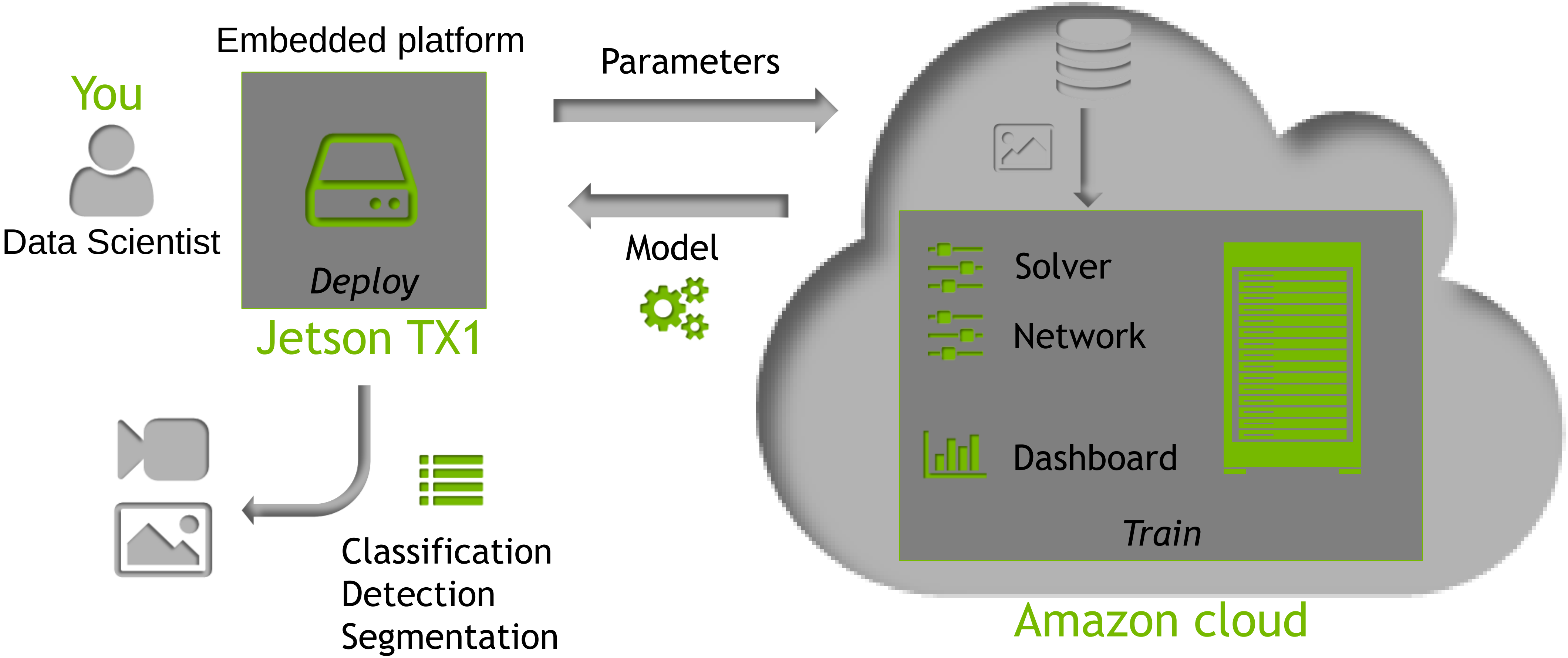
MULTI-GPU

# TODAY

# DEEP LEARNING ON GPUS: AN END-TO-END SOLUTION



# TODAY'S LAB



Embedded platform

You

Data Scientist

*Deploy*

Jetson TX1

Parameters

Model

Solver

Network

Dashboard

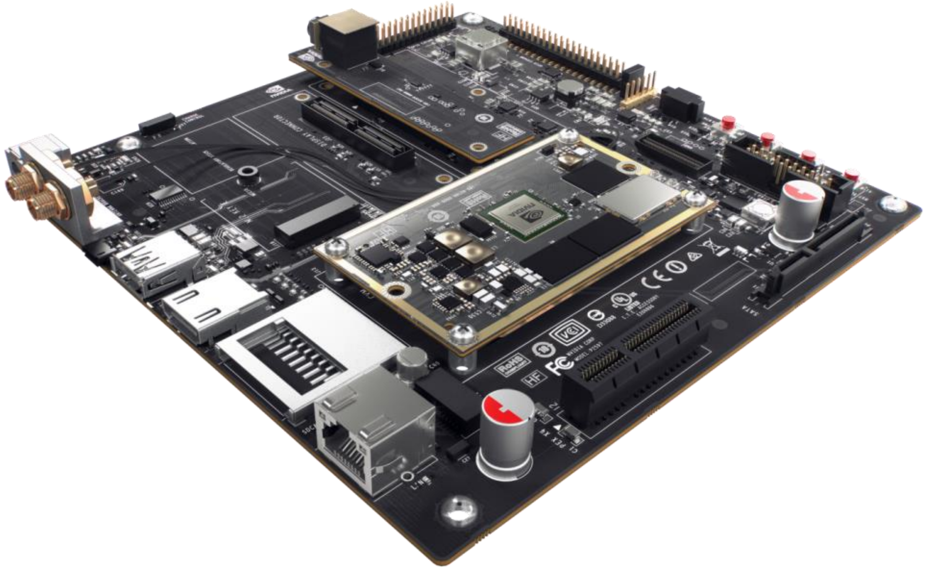
*Train*

Amazon cloud

Classification
Detection
Segmentation

# EMBEDDED DEPLOYMENT

## On Jetson TX1



Inference at 258 img/s

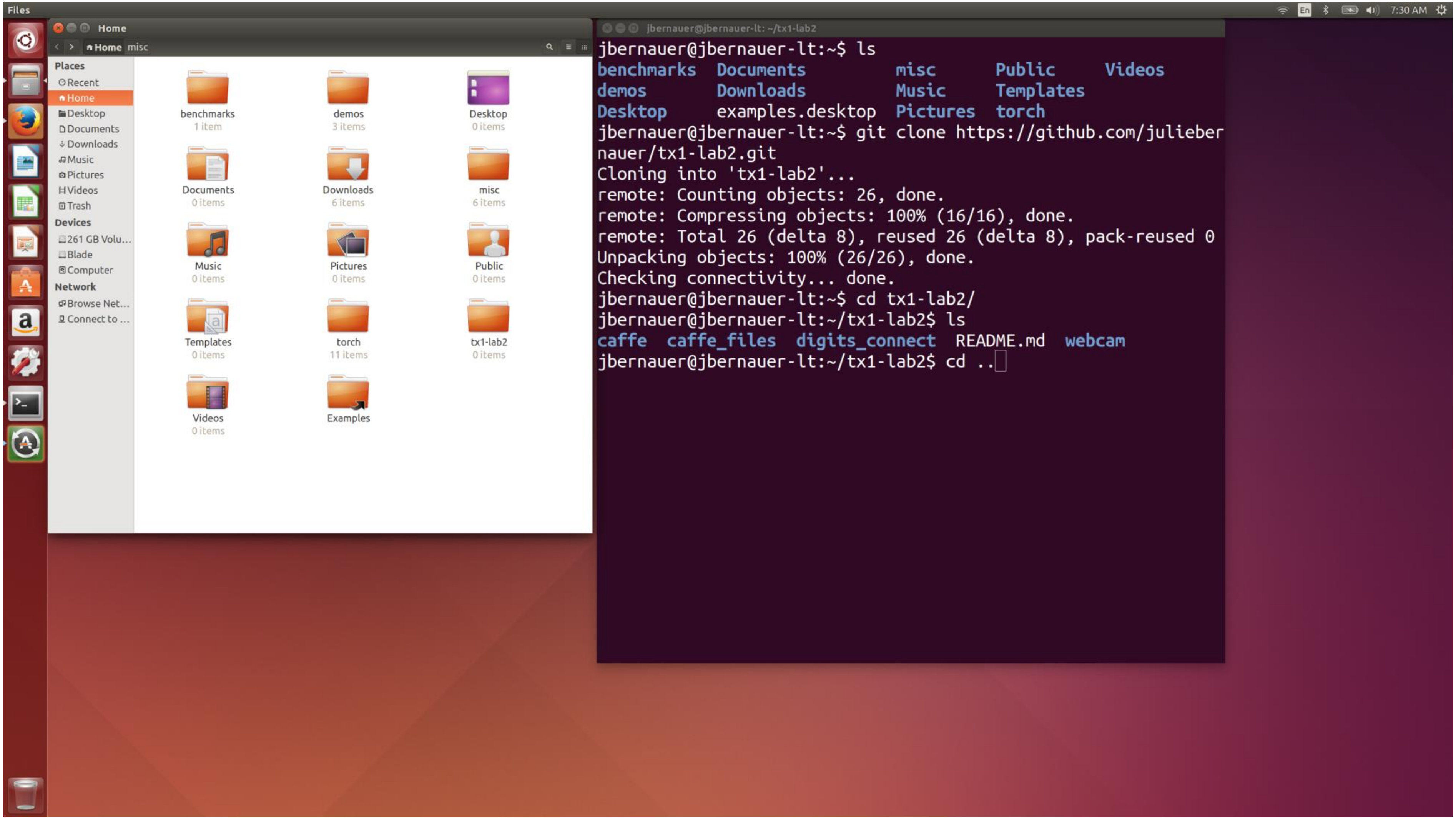
No need to change code

Simply compile Caffe and copy a trained `.caffemodel` to TX1

# SET UP

# USING THE TERMINAL

## Linux Ubuntu 14.04



```
jbernauer@jbernauer-lt:~$ ls
benchmarks  Documents         misc      Public     Videos
demos       Downloads         Music     Templates
Desktop     examples.desktop  Pictures  torch
jbernauer@jbernauer-lt:~$ git clone https://github.com/julieber
nauer/tx1-lab2.git
Cloning into 'tx1-lab2'...
remote: Counting objects: 26, done.
remote: Compressing objects: 100% (16/16), done.
remote: Total 26 (delta 8), reused 26 (delta 8), pack-reused 0
Unpacking objects: 100% (26/26), done.
Checking connectivity... done.
jbernauer@jbernauer-lt:~$ cd tx1-lab2/
jbernauer@jbernauer-lt:~/tx1-lab2$ ls
caffe  caffe_files  digits_connect  README.md  webcam
jbernauer@jbernauer-lt:~/tx1-lab2$ cd ..
```

# 0- DOWNLOADING LAB DATA

## From the TX1

- Install git on the tx1: *sudo apt-get install git*
- Data for the TX1 is on github, in the terminal, type:

  *git clone https://github.com/juliebernauer/tx1-lab2.git*

- The instructions can also be seen on the github page:

  https://github.com/juliebernauer/tx1-lab2

# 1- CONNECTING TO THE AMAZON CLOUD

## From the TX1

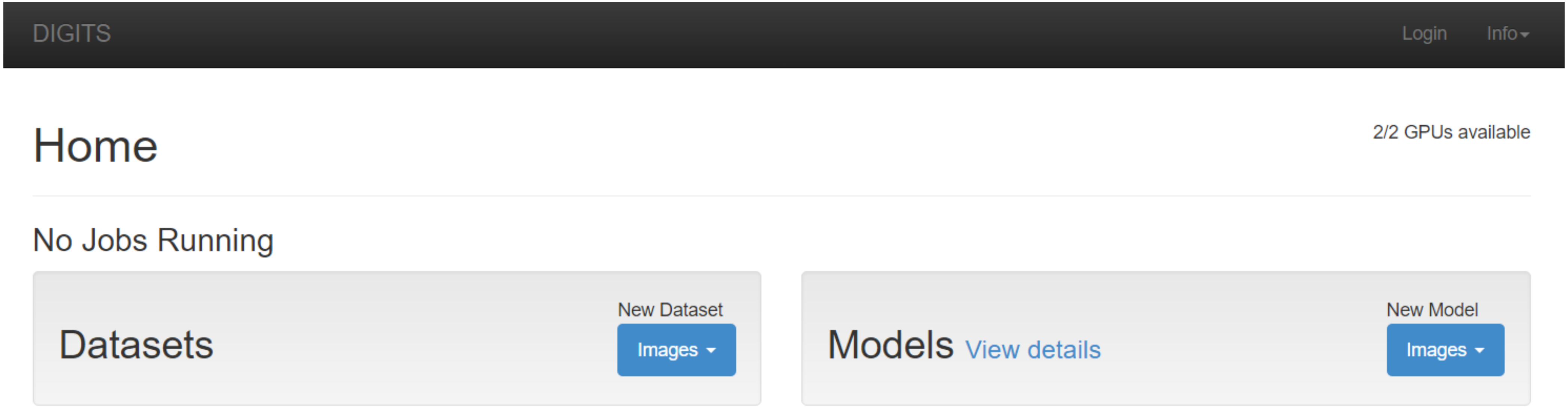
- Connect to https://nvlabs.qwiklabs.com
- Pick the *Digits and Jetson TX1* lab, click the *Start* button.
- Wait for the lab to launch
- Click on *Connection -> Start your lab*
- Click on the DIGITS link

# GETTING STARTED WITH DEEP LEARNING DIGITS

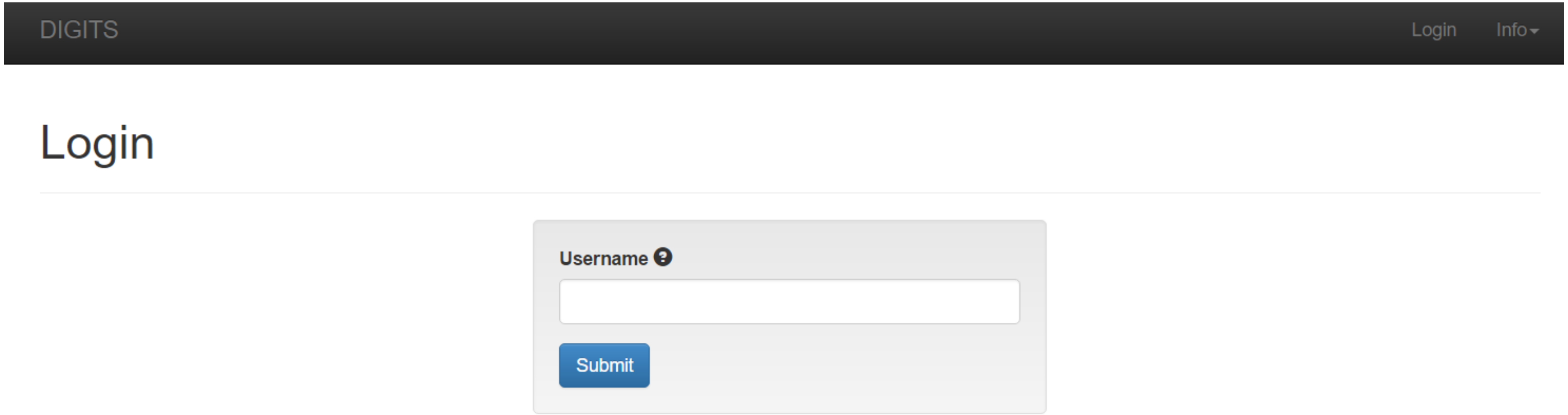
# 2- TRAIN AN IMAGE CLASSIFICATION MODEL

## Using digits on Amazon EC2

In the web browser, connect to your amazon instance DIGITS web server



# CREATE YOUR USERNAME

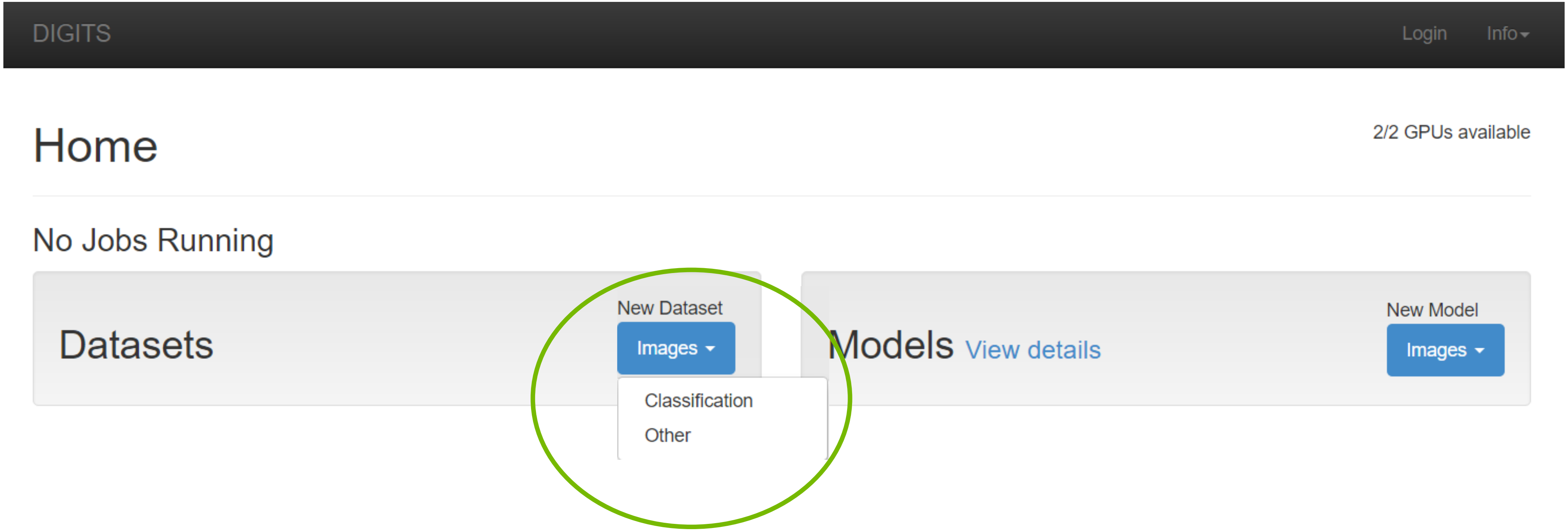


## Login

Username ?

Submit

# CREATE YOUR DATASET



2/2 GPUs available

No Jobs Running

# CREATE A DATASET

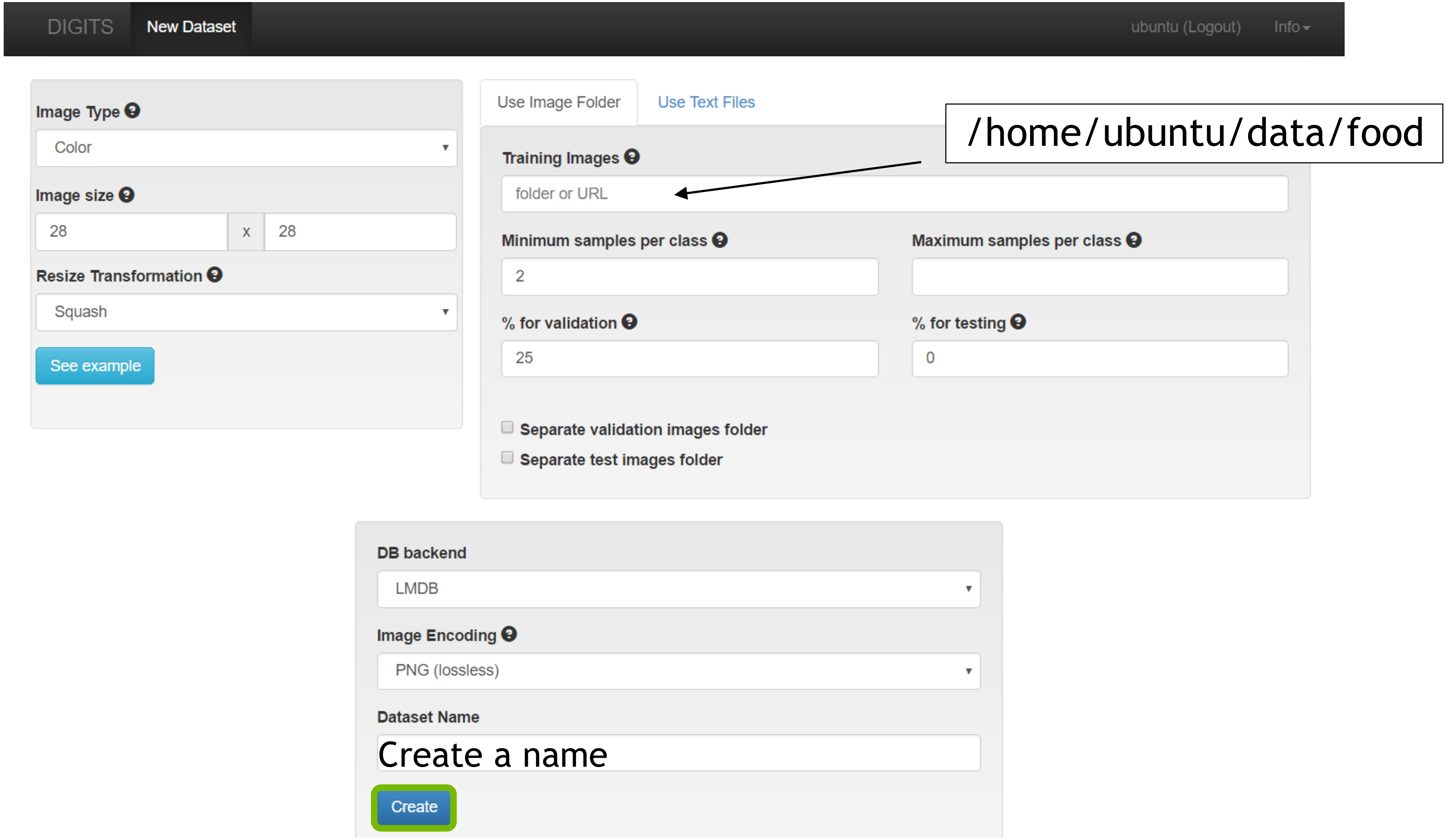


Image Type: Color

Image size: 28 x 28

Resize Transformation: Squash

Training Images: folder or URL

/home/ubuntu/data/food

Minimum samples per class: 2

Maximum samples per class:

% for validation: 25

% for testing: 0

□ Separate validation images folder

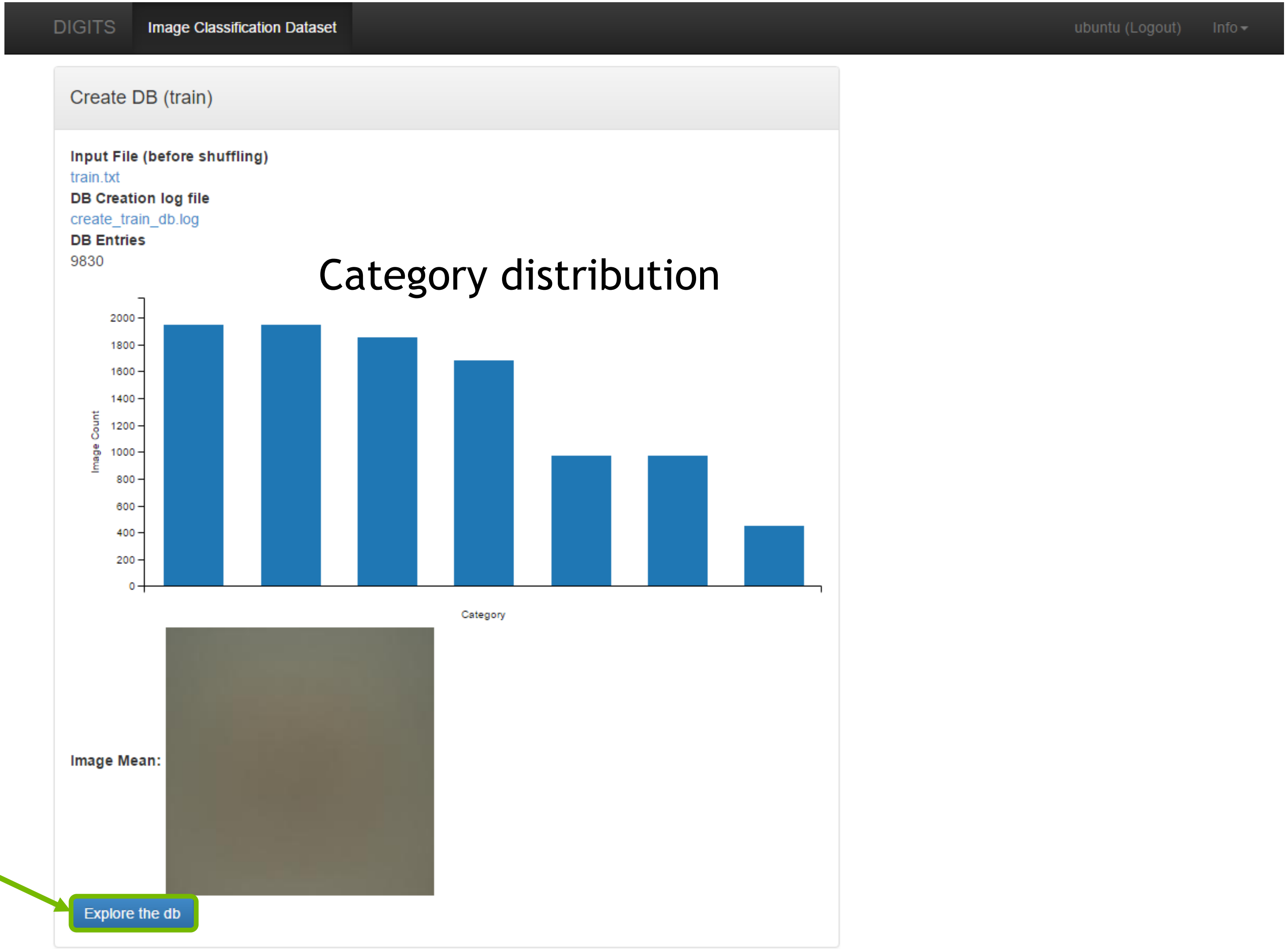
□ Separate test images folder

DB backend: LMDB

Image Encoding: PNG (lossless)

Dataset Name: Create a name

# EVALUATE YOUR DATASET



Create DB (train)

**Input File (before shuffling)**
train.txt

**DB Creation log file**
create_train_db.log

**DB Entries**
9830

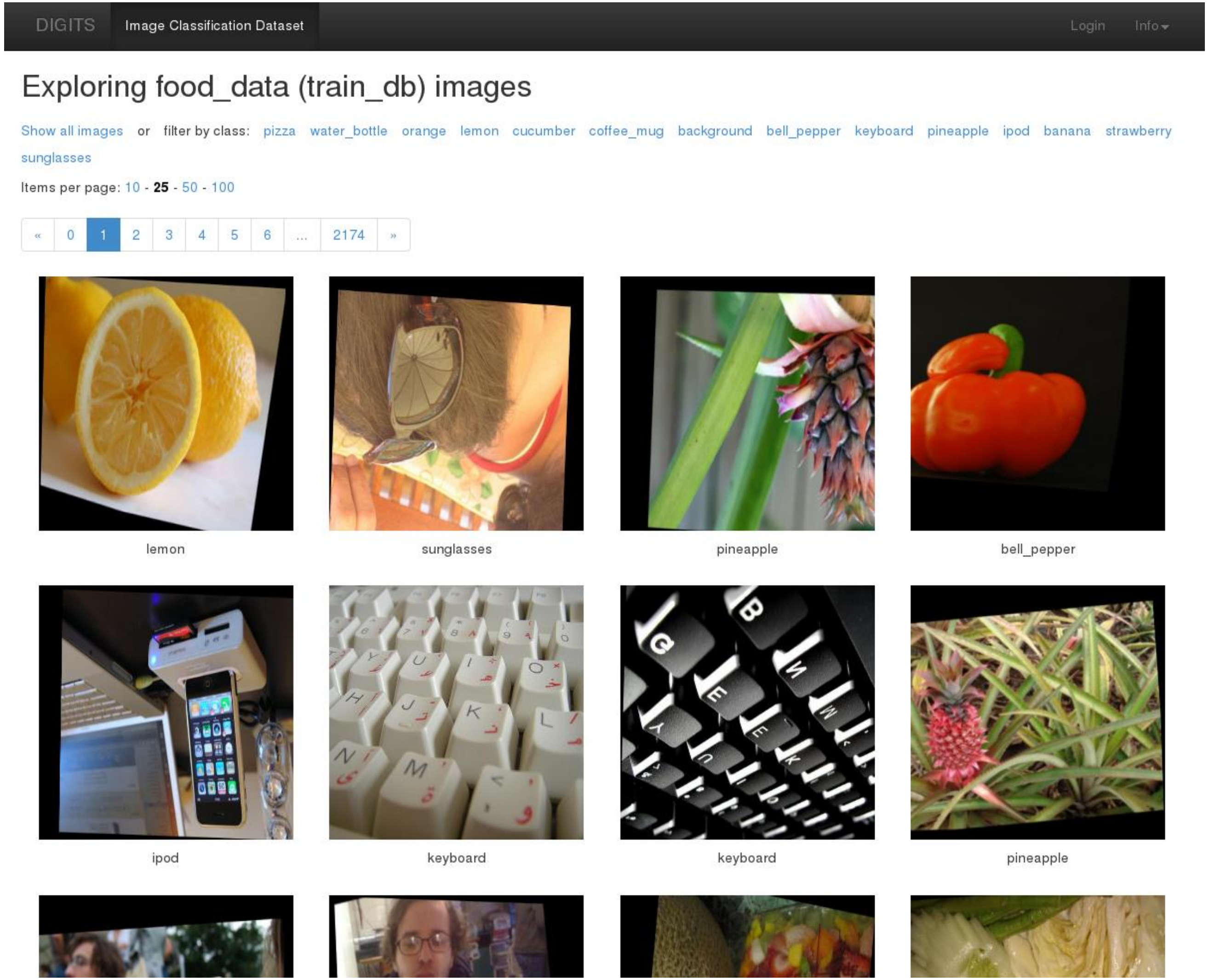
Category distribution

**Image Mean:**

Explore your dataset

# EXPLORE YOUR DATASET

Pick the *food_data* dataset

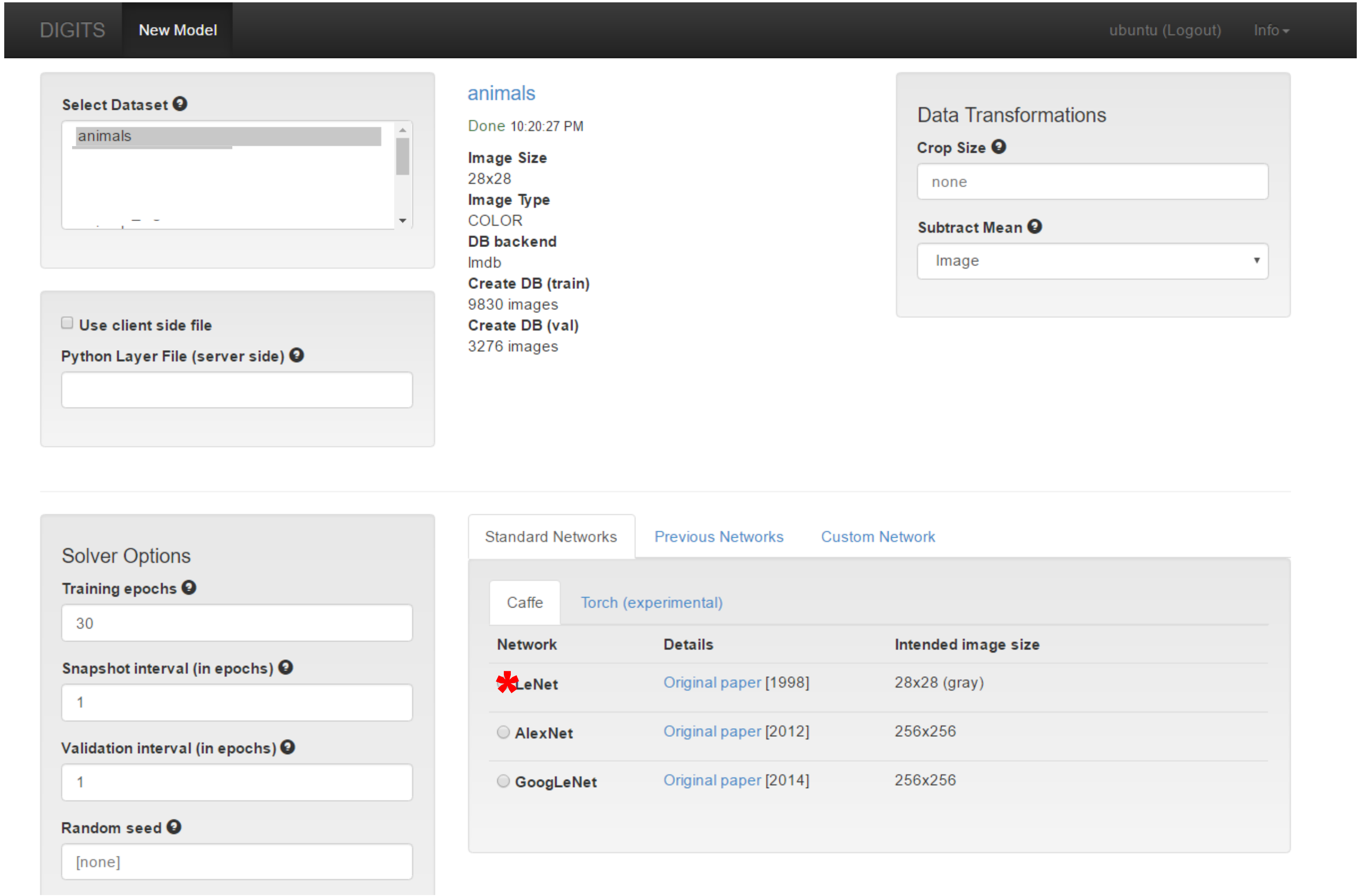


Exploring food_data (train_db) images

Show all images or filter by class: pizza water_bottle orange lemon cucumber coffee_mug background bell_pepper keyboard pineapple ipod banana strawberry sunglasses

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

# TRAIN A NETWORK



**Select Dataset**

animals

□ **Use client side file**

**Python Layer File (server side)**

animals

Done 10:20:27 PM

**Image Size**
28x28
**Image Type**
COLOR
**DB backend**
lmdb
**Create DB (train)**
9830 images
**Create DB (val)**
3276 images

Data Transformations

**Crop Size**
none

**Subtract Mean**
Image

Solver Options

**Training epochs**
30

**Snapshot interval (in epochs)**
1

**Validation interval (in epochs)**
1

**Random seed**
[none]

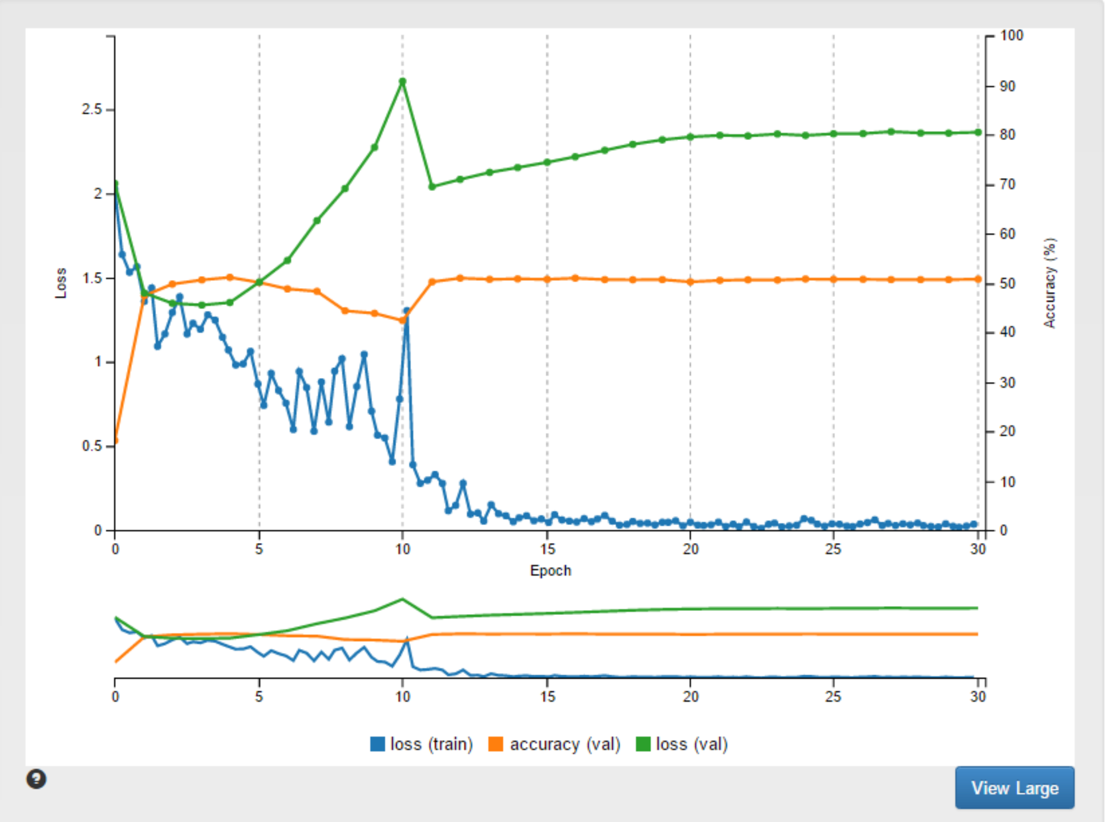
Standard Networks | Previous Networks | Custom Network

Caffe | Torch (experimental)

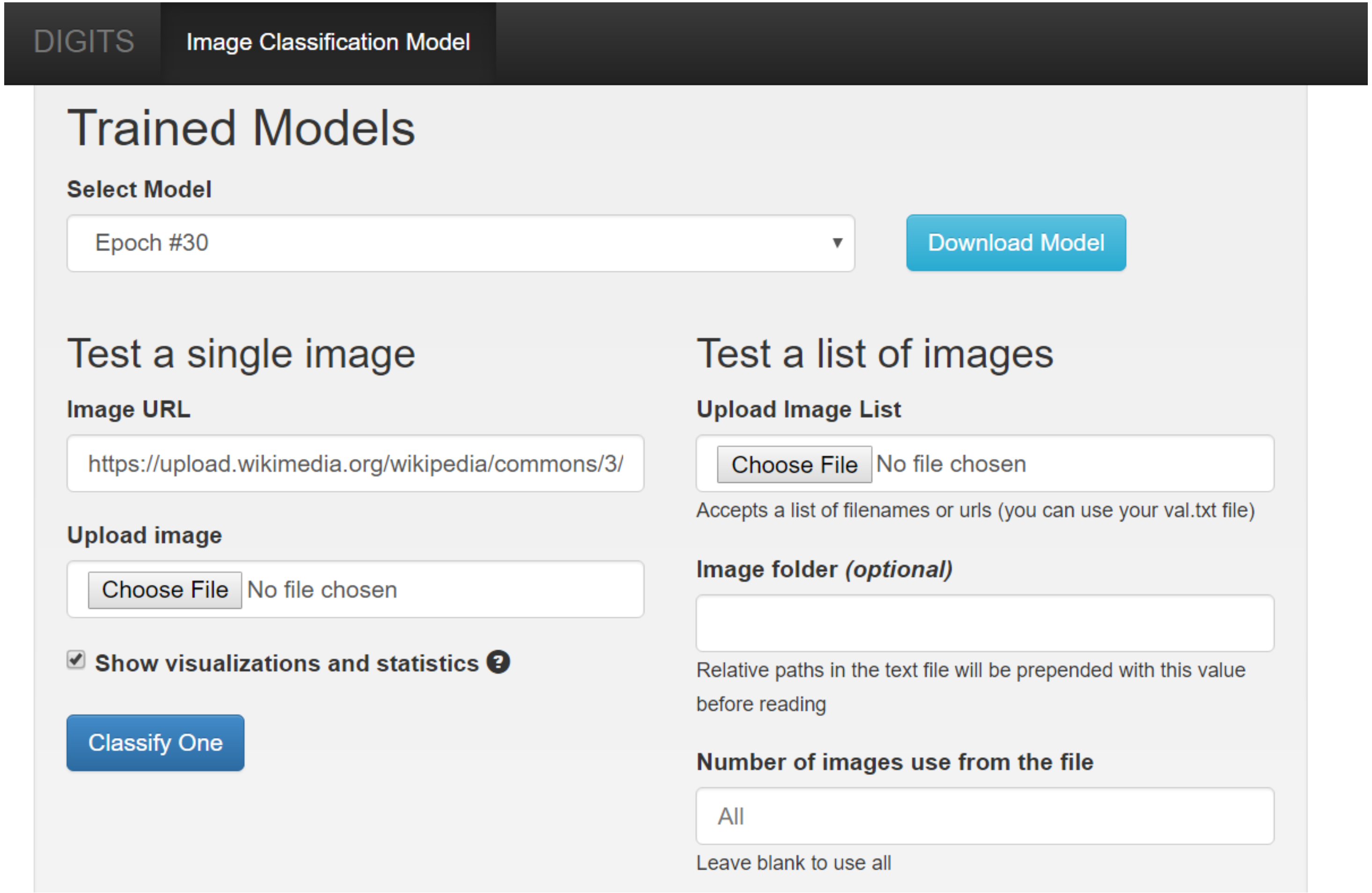
| Network | Details | Intended image size |
|---|---|---|
| * LeNet | Original paper [1998] | 28x28 (gray) |
| ○ AlexNet | Original paper [2012] | 256x256 |
| ○ GoogLeNet | Original paper [2014] | 256x256 |

# REVIEW RESULTS

- What is your accuracy?
- How long did it take to train?



# CLASSIFY AN IMAGE



## Trained Models

**Select Model**

Epoch #30 ▾

### Test a single image

**Image URL**

https://upload.wikimedia.org/wikipedia/commons/3/

**Upload image**

Choose File No file chosen

☑ **Show visualizations and statistics** ?

### Test a list of images

**Upload Image List**

Choose File No file chosen

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

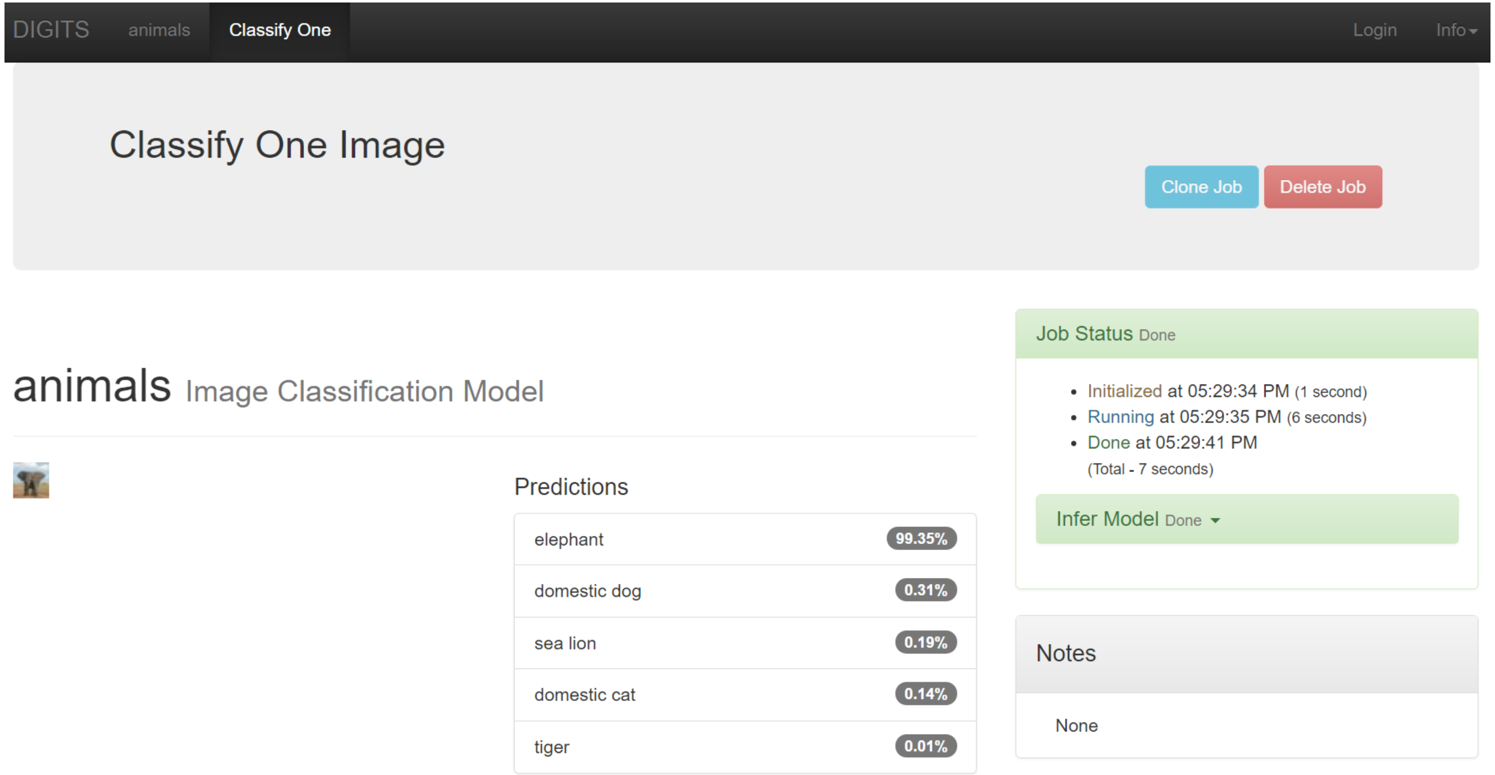
**Number of images use from the file**

All

Leave blank to use all

# CLASSIFY AN IMAGE

## Explore Network Response



Classify One Image

animals Image Classification Model

Predictions

|  |  |
|---|---|
| elephant | 99.35% |
| domestic dog | 0.31% |
| sea lion | 0.19% |
| domestic cat | 0.14% |
| tiger | 0.01% |

Job Status Done

- Initialized at 05:29:34 PM (1 second)
- Running at 05:29:35 PM (6 seconds)
- Done at 05:29:41 PM
  (Total - 7 seconds)

Infer Model Done

Notes

None

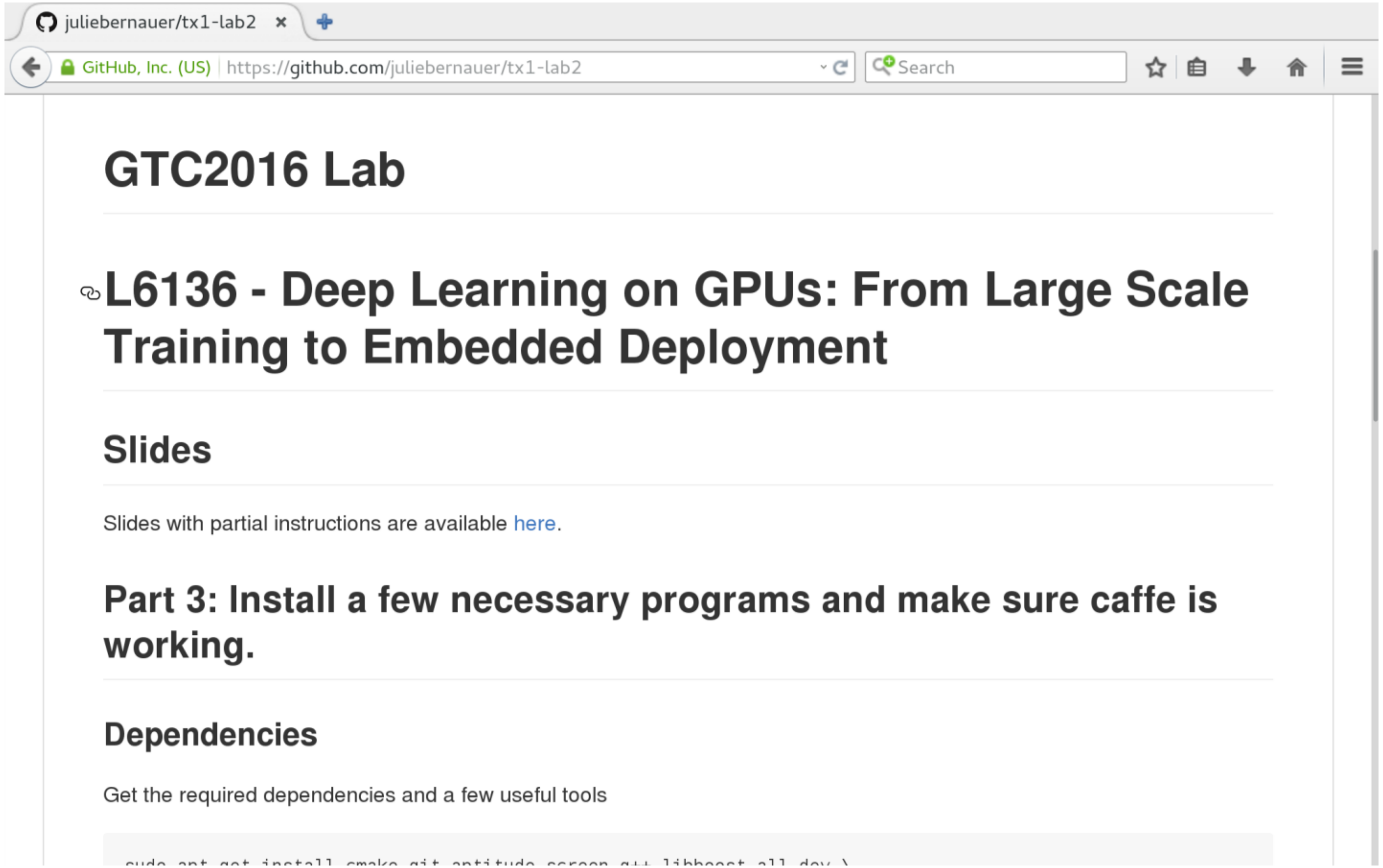
Scroll down to view your network's response

# DEPLOYMENT ON TX1

# 3- CAFFE

## TX1

Follow the instructions on the Github page



# GTC2016 Lab

## L6136 - Deep Learning on GPUs: From Large Scale Training to Embedded Deployment

### Slides

Slides with partial instructions are available here.

### Part 3: Install a few necessary programs and make sure caffe is working.

#### Dependencies

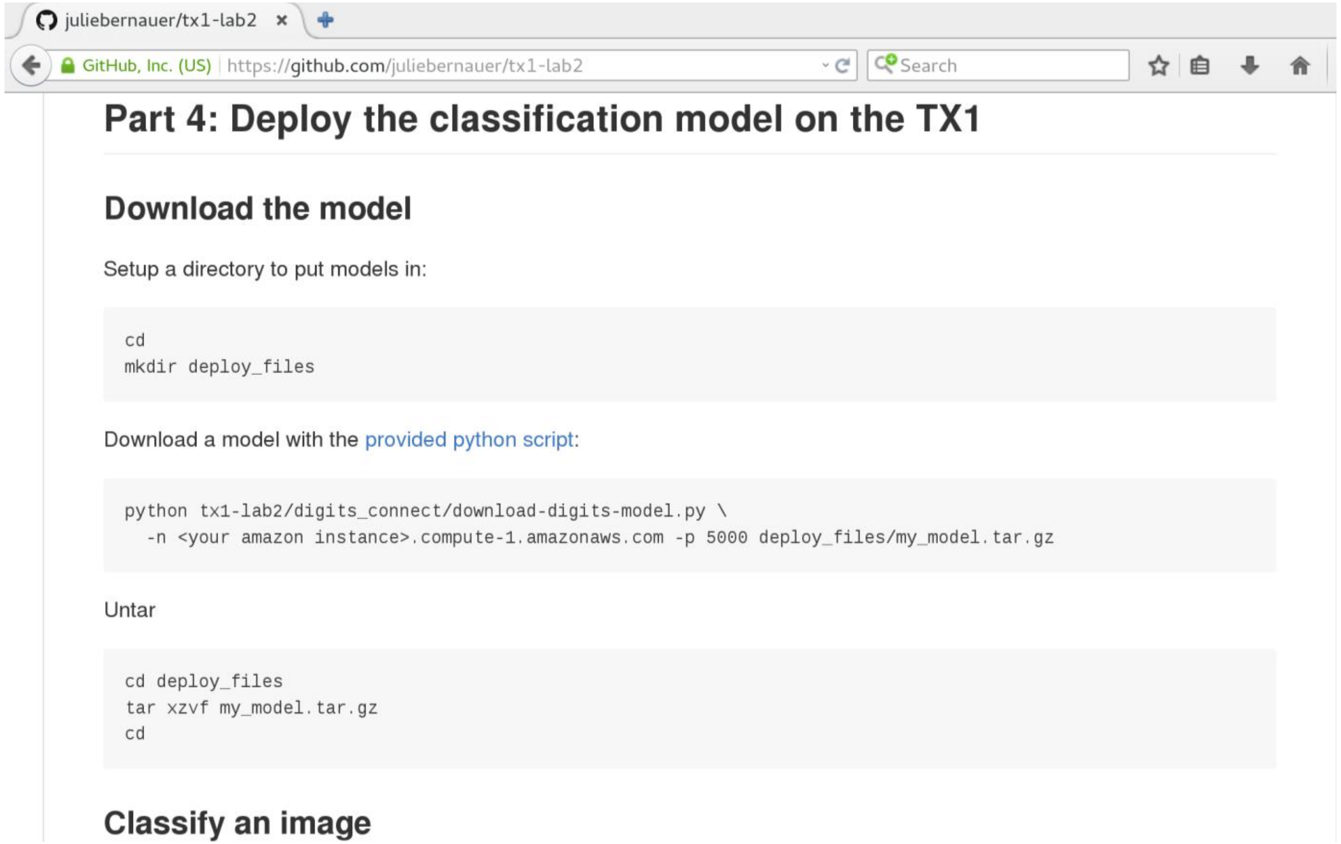
Get the required dependencies and a few useful tools

```
sudo apt-get install cmake git aptitude screen g++ libboost-all-dev \
```

# 4- DEPLOY THE IMAGE CLASSIFICATION MODEL

TX1

Follow the instructions on the Github page



## Part 4: Deploy the classification model on the TX1

### Download the model

Setup a directory to put models in:

```
cd
mkdir deploy_files
```

Download a model with the provided python script:

```
python tx1-lab2/digits_connect/download-digits-model.py \
  -n <your amazon instance>.compute-1.amazonaws.com -p 5000 deploy_files/my_model.tar.gz
```

Untar

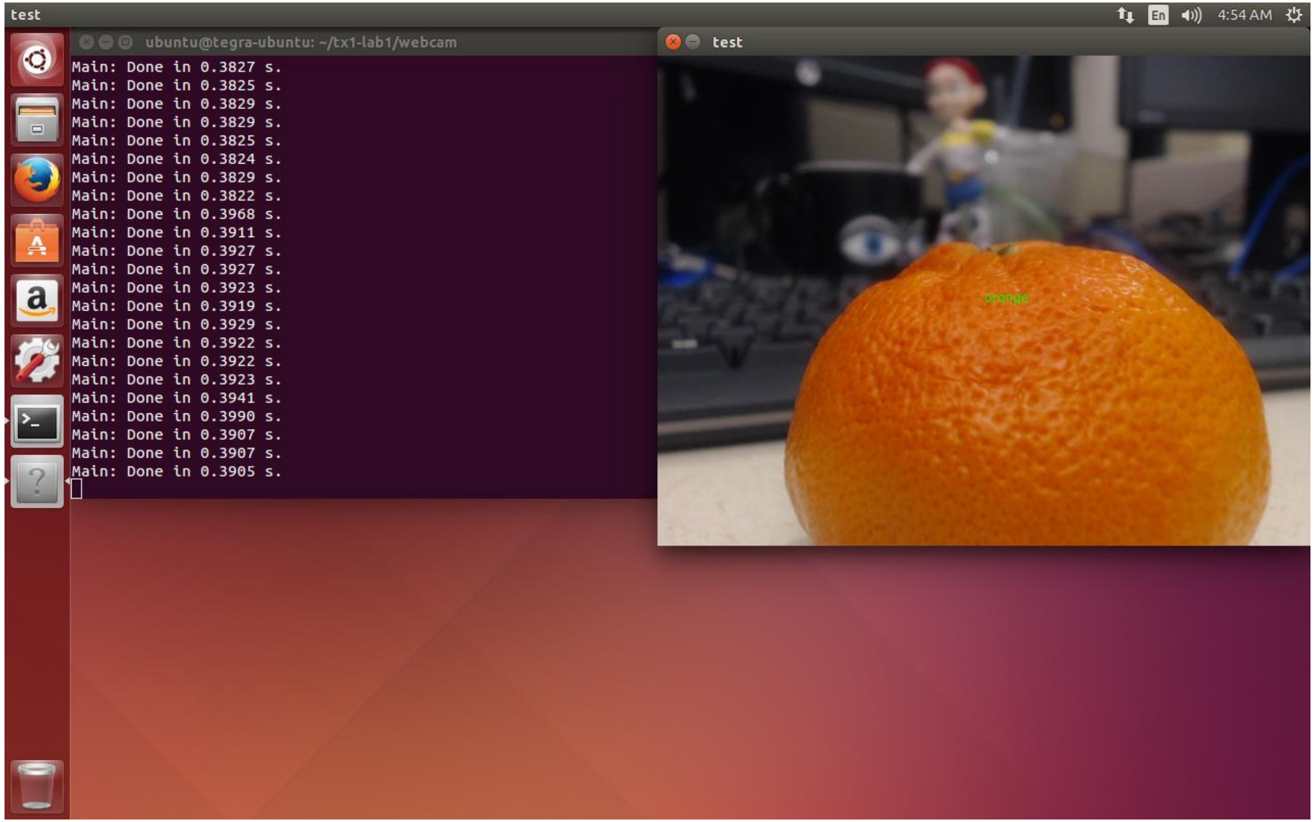
```
cd deploy_files
tar xzvf my_model.tar.gz
cd
```

### Classify an image

# 5- FUN WITH THE WEBCAM

## TX1

Follow the instructions on the Github page



```
Main: Done in 0.3827 s.
Main: Done in 0.3825 s.
Main: Done in 0.3829 s.
Main: Done in 0.3829 s.
Main: Done in 0.3825 s.
Main: Done in 0.3824 s.
Main: Done in 0.3829 s.
Main: Done in 0.3822 s.
Main: Done in 0.3968 s.
Main: Done in 0.3911 s.
Main: Done in 0.3927 s.
Main: Done in 0.3927 s.
Main: Done in 0.3923 s.
Main: Done in 0.3919 s.
Main: Done in 0.3929 s.
Main: Done in 0.3922 s.
Main: Done in 0.3922 s.
Main: Done in 0.3923 s.
Main: Done in 0.3941 s.
Main: Done in 0.3990 s.
Main: Done in 0.3907 s.
Main: Done in 0.3907 s.
Main: Done in 0.3905 s.
```

GPU TECHNOLOGY CONFERENCE

April 4-7, 2016 | Silicon Valley

# THANK YOU

JOIN THE CONVERSATION

#GTC16

JOIN THE NVIDIA DEVELOPER PROGRAM AT developer.nvidia.com/join

PRESENTED BY

NVIDIA.