

Agenda:

1. Short introduction to AlexNet, Convolutional Neural Network, trained by multiple GPUs.
2. Practical exercise using NVIDIA DIGITS on Amazon Web Services (AWS).

By:

Knut Berg Kaldestad &  
Geir Hovland



# Cloud- and GPU-Based Machine Learning

2017-03-15

E-mail: [sfi@mechatronics.no](mailto:sfi@mechatronics.no)

<https://sfi.mechatronics.no>

AlexNet 2012, University of Toronto

<https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>

## ImageNet Classification with Deep Convolutional Neural Networks

Part of: [Advances in Neural Information Processing Systems 25 \(NIPS 2012\)](#)

[\[PDF\]](#) [\[BibTeX\]](#) [\[Supplemental\]](#)

### Authors

- [Alex Krizhevsky](#)
- [Ilya Sutskever](#)
- [Geoffrey E. Hinton](#)

### Abstract

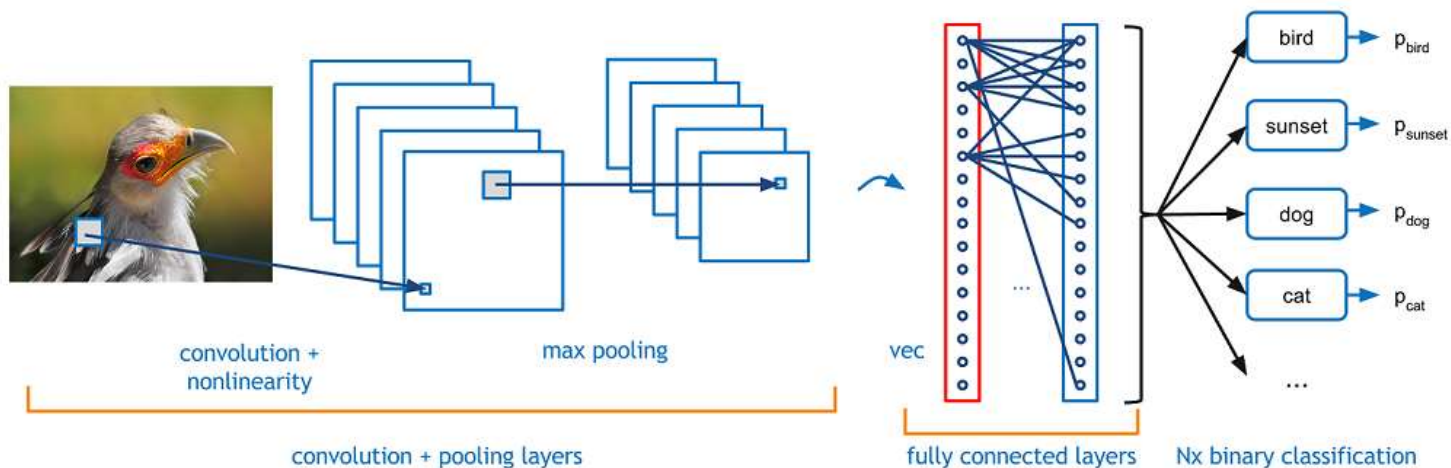
We trained a large, deep convolutional neural network to classify the 1.3 million high-resolution images in the LSVRC-2010 ImageNet training set into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 39.7% and 18.9% which is considerably better than the previous state-of-the-art results. The neural network, which has 60 million parameters and 500,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and two globally connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of convolutional nets. To reduce overfitting in the globally connected layers we employed a new regularization method that proved to be very effective.

# A Beginner's Guide To Understanding Convolutional Neural Networks

<https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks>

## Biological Connection

But first, a little background. When you first heard of the term convolutional neural networks, you may have thought of something related to neuroscience or biology, and you would be right. Sort of. CNNs do take a biological inspiration from the visual cortex. The visual cortex has small regions of cells that are sensitive to specific regions of the visual field. This idea was expanded upon by a fascinating experiment by Hubel and Wiesel in 1962 ([Video](#)) where they showed that some individual neuronal cells in the brain responded (or fired) only in the presence of edges of a certain orientation. For example, some neurons fired when exposed to vertical edges and some when shown horizontal or diagonal edges. Hubel and Wiesel found out that all of these neurons were organized in a columnar architecture and that together, they were able to produce visual perception. This idea of specialized components inside of a system having specific tasks (the neuronal cells in the visual cortex looking for specific characteristics) is one that machines use as well, and is the basis behind CNNs.



## AlexNet: Training Using Multiple GPUs.

- 8 layers with weights
- First five are convolutional, the last three are fully connected
- Final output: 1000 class labels
- Layers 2, 4 and 5 receive input only from same GPU

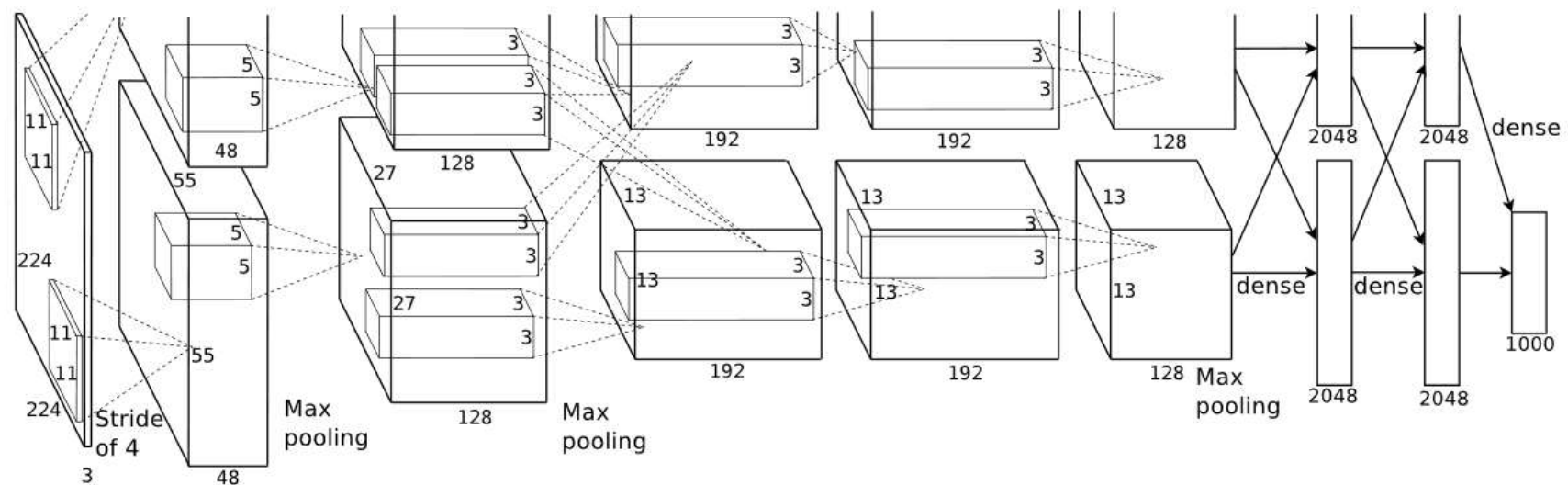


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

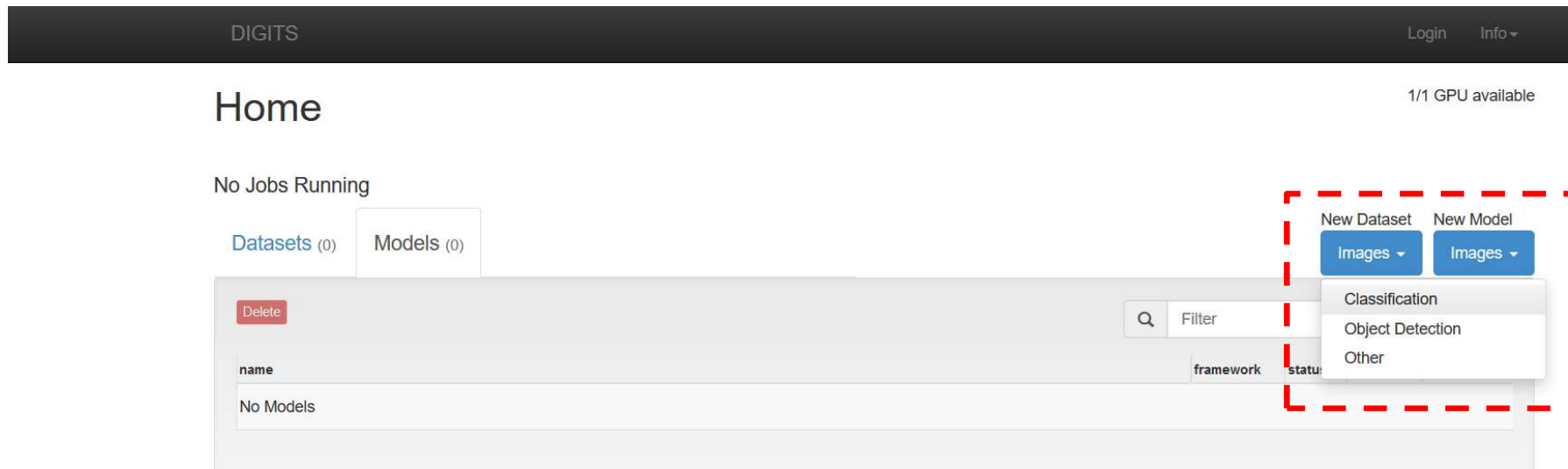


## NVIDIA DIGITS on Amazon Web Services (AWS)

Go to website:

PS: Make sure that you have the latest Adobe Flash Plugin for your browser.

Select: «New Dataset» – «Images» – «Classification»



The screenshot shows the NVIDIA DIGITS web interface. At the top is a dark header with the word "DIGITS" on the left and "Login" and "Info" on the right. Below the header, the word "Home" is displayed on the left, and "1/1 GPU available" is on the right. The main content area shows "No Jobs Running" and two tabs: "Datasets (0)" and "Models (0)". Below the tabs is a table with columns for "name", "framework", and "status". The table is currently empty, showing "No Models". To the right of the table, there is a "New Dataset" button with a dropdown menu open, showing "Images" selected. Below "Images", there is a "Classification" option. A red dashed box highlights the "New Dataset" button, the "Images" dropdown, and the "Classification" option.

## Username and Dataset

Must start with a lowercase letter

Username ?

ola

Submit

Image Type ?

Color

Image size (Width x Height) ?

300

200

Resize Transformation ?

Squash

See example

Use Image Folder

Use Text Files

Training Images ?

/home/ubuntu/3\_ObjCat

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

myname

Create

Click

Wait until dataset is generated (100%)

Create DB (train) Running ▾

12%

Estimated time remaining: 1 minute, 42 seconds

- Initialized at 12:21:11 PM (2 seconds)
- Running at 12:21:13 PM

Create DB (val) Running ▾

40%

Estimated time remaining: 20 seconds

- Initialized at 12:21:11 PM (2 seconds)
- Running at 12:21:13 PM

## Dataset Generated: Job Information (Page 1)

### Job Information

**Job Directory**

/usr/share/digits/digits/jobs/20170312-064215-e171

**Image Dimensions**

300x200 (Width x Height)

**Image Type**

Color

**Resize Transformation**

Squash

**DB Backend**

Imdb

**Image Encoding**

png

**DB Compression**

none

**Dataset size**

0 B

### Parse Folder (train/val)

**Folder**

/home/ubuntu/3\_ObjCat

### Job Status Done

- **Initialized** at 06:42:15 AM (1 second)
  - **Running** at 06:42:16 AM (7 seconds)
  - **Done** at 06:42:23 AM
- (Total - 8 seconds)

### Parse Folder (train/val) Done

- **Initialized** at 06:42:15 AM (1 second)
  - **Running** at 06:42:17 AM (1 second)
  - **Done** at 06:42:18 AM
- (Total - 2 seconds)

### Create DB (train) Done

### Create DB (val) Done

### Create DB (train)

**Input File (before shuffling)**

[train.txt](#)

**DB Creation log file**

[create\\_train\\_db.log](#)

### Notes

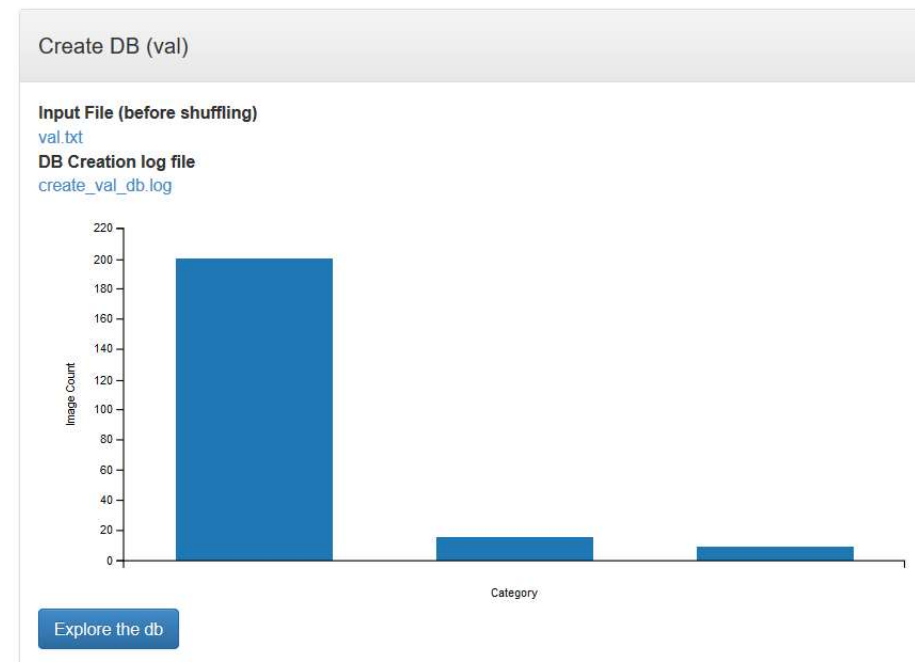
None [🔗](#)



## Dataset: Image Count vs Category

Left: Training Set

Right: Validation Set



## Explore Dataset: 3 categories (airplanes, elephants, strawberries)

### Exploring myname (train\_db) images

Show all images or filter by class: [airplanes](#) [elephant](#) [strawberry](#)

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

« 0 1 2 3 4 5 ... 26 »



airplanes



airplanes



airplanes



airplanes



strawberry



airplanes



airplanes



airplanes



airplanes



elephant



airplanes



airplanes

## Back to DIGITS: Start Screen

Click here to return home



### Home

1/1 GPU available

No Jobs Running

Datasets (1)

Models (0)

The main content area of the DIGITS interface. It shows a "Delete" button, a search bar with "Filter", and a table with columns "name", "framework", "status", "elapsed", and "size". The table is currently empty, displaying "No Models". On the right, there are two buttons: "New Dataset" and "New Model", both with "Images" dropdown menus. A red dashed box highlights the "New Model" dropdown menu, which is open, showing options: "Classification", "Object Detection", and "Other". A blue arrow points from the text "Select «Classification»" to the "Classification" option.

Select «Classification»

You now have a dataset defined.

Select Dataset: myname – must match name you created.

Select Dataset ?

myname

3 Categories

myname

Done 08:42:23 AM

Image Size  
300x200

Image Type  
COLOR

DB backend  
Imdb

Create DB (train)  
674 images

Create DB (val)  
225 images

Python Layers ?

Server-side file ?

☐ Use client-side file

Solver Options

Training epochs ?

15

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

Crop Size ?

none

Subtract Mean ?

Image

## Selection of Network Type

Standard Networks Previous Networks Custom Network

Caffe Torch

Network	Details	Intended image size
<input type="radio"/> LeNet	<a href="#">Original paper [1998]</a>	28x28 (gray)
<input checked="" type="radio"/> AlexNet	<a href="#">Original paper [2012]</a>	256x256 <a href="#">Customize</a>
<input type="radio"/> GoogLeNet	<a href="#">Original paper [2014]</a>	256x256

Model Name ?

Create

Click



## Training the neural network: 30 Epochs

**Job Directory**  
 /usr/share/digits/digits  
 /jobs/20170312-065914-ea1b  
**Disk Size**  
 0 B  
**Network (train/val)**  
[train\\_val.prototxt](#)  
**Network (deploy)**  
[deploy.prototxt](#)  
**Solver**  
[solver.prototxt](#)  
**Raw caffe output**  
[caffe\\_output.log](#)

### Dataset

[myname](#)  
 Done 06:42:23 AM  
**Image Size**  
 300x200  
**Image Type**  
 COLOR  
**DB backend**  
 Imdb  
**Create DB (train)**  
 674 images  
**Create DB (val)**  
 225 images

### Job Status Initialized

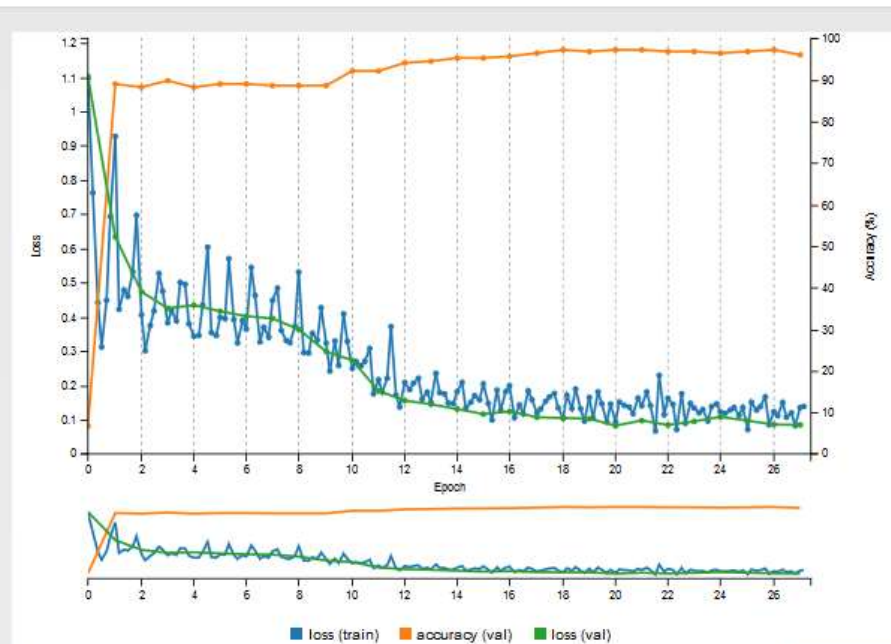
- Initialized at 06:59:14 AM

Train Caffe Model Initialized ▾

## GPU Usage

### GRID K520 (#0)

**Memory**  
 2.38 GB / 4 GB (59.5%)  
**GPU Utilization**  
 99%  
**Temperature**  
 75 °C



[View Large](#)

### Related jobs

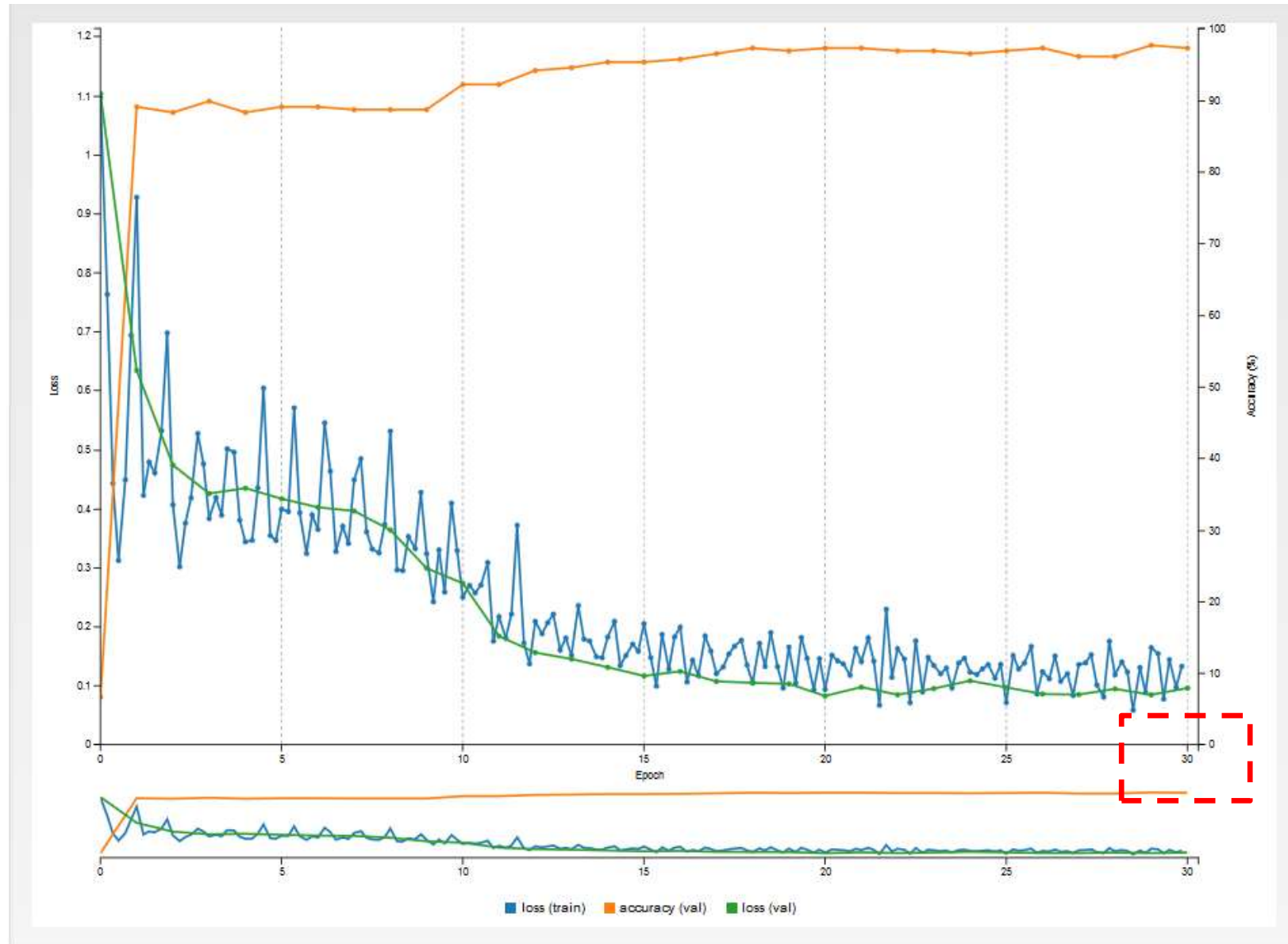
Image Classification Dataset

[myname](#) Done ▾

### Notes

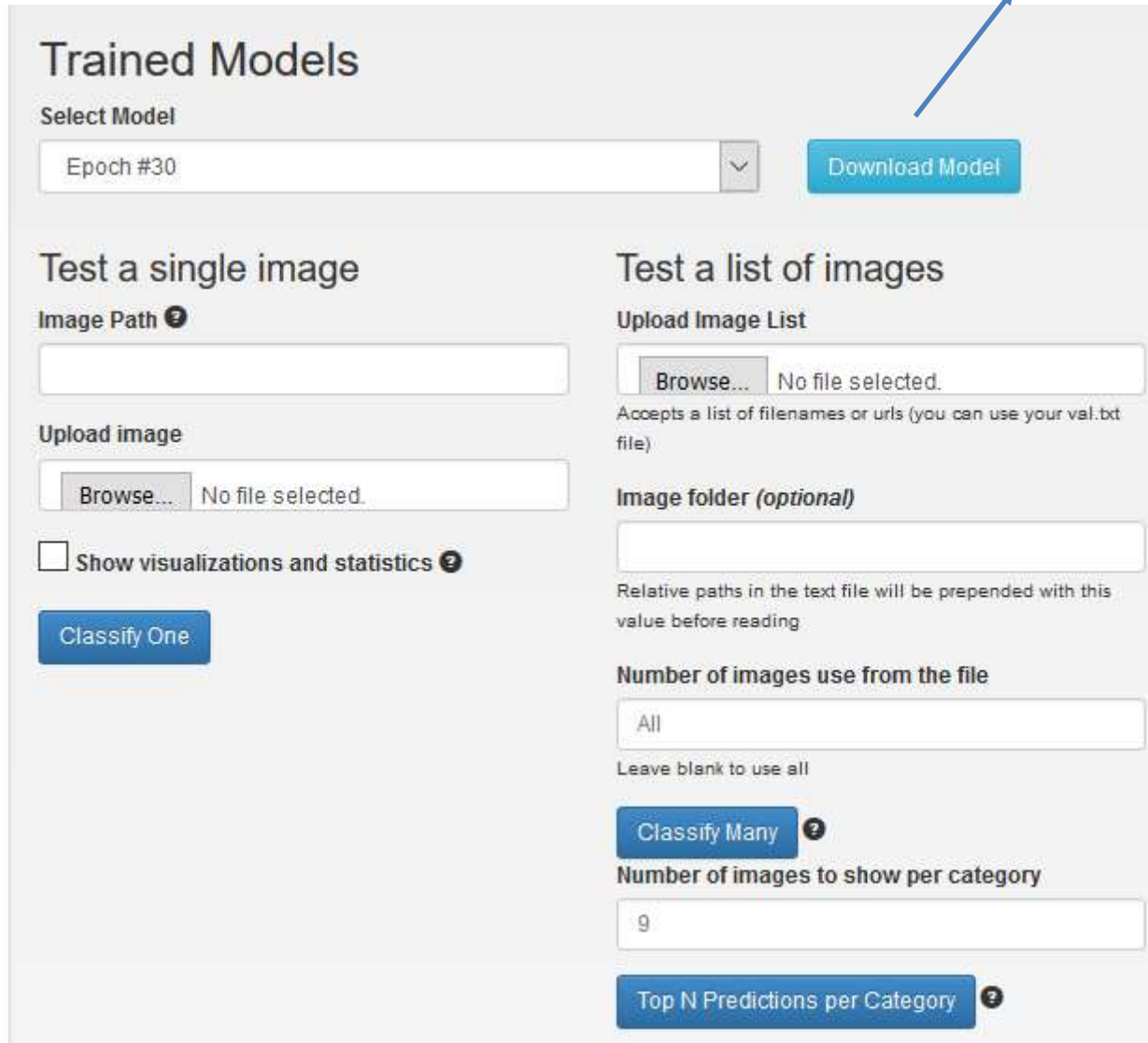
None [✎](#)

Final result: accuracy about 96-97%



## Trained Models

The model can be downloaded  
for offline use (about 200Mb)  
Not necessary at conference.



### Trained Models

Select Model

Epoch #30 ▼ Download Model

#### Test a single image

Image Path ?

Upload image

Browse... No file selected.

☐ Show visualizations and statistics ?

Classify One

#### Test a list of images

Upload Image List

Browse... No file selected.

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 Many ?

Number of images to show per category

9

Top N Predictions per Category ?

Test a single image: /home/ubuntu/3\_ObjCat/strawberry/image\_0001.jpg

## Trained Models

Select Model

Epoch #30

Download Model

### Test a single image

Image Path ?

/home/ubuntu/3\_ObjCat/strawberry/image\_0001

Upload image

Browse... No file selected.

☐ Show visualizations and statistics ?

Classify One

### Test a list of images

Upload Image List

Browse... No file selected.

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 Many ?

Number of images to show per category

9

Top N Predictions per Category ?

Click

## Image Classification Example: 91.5% confidence

myname Image Classification Model



### Predictions

strawberry	91.51%
elephant	7.13%
airplanes	1.36%

Job Status: Done

- Initialized at 07:13:09 AM (1 second)
  - Running at 07:13:10 AM (4 seconds)
  - Done at 07:13:15 AM
- (Total - 5 seconds)

Infer Model Done ▾

### Notes

None



Random picture from the internet: 79.5% confidence

<http://www.myelomacrowd.org/wp-content/uploads/2015/10/elephant.jpg>

Test a single image

Image Path <sup>?</sup>

[vd.org/wp-content/uploads/2015/10/elephant.jpg](http://www.myelomacrowd.org/wp-content/uploads/2015/10/elephant.jpg)

Upload image

No file selected.

☐ Show visualizations and statistics <sup>?</sup>

Click

myname Image Classification Model



Predictions

elephant	79.51%
strawberry	13.16%
airplanes	7.33%

Job Status Done

- Initialized at 07:23:21 AM (1 second)
  - Running at 07:23:22 AM (5 seconds)
  - Done at 07:23:28 AM
- (Total - 6 seconds)

Infer Model Done ▾

Notes

None

## Summary and Conclusions

- Practical Demonstration of Deep Learning Framework
- DIGITS puts the power of Deep Learning into the hands of engineers and scientists who are not experts of computer science and GPU programming
- Results can be created relatively quickly, as demonstrated by this conference exercise
- Trained models from the cloud (AWS) can be downloaded and used offline without access to the internet
- Open source: <https://developer.nvidia.com/digits>