



AI Immersion Workshop

May 9, 2017 | W Seattle Hotel, Seattle, WA





De-mystifying Deep Learning

Anusua Trivedi
Data Scientist
Email: antriv@microsoft.com
Twitter: [@anurive](https://twitter.com/anurive)



Talk Outline

- ❑ Deep Learning (DL)
- ❑ Deep Neural Networks (DNN)
- ❑ Types of DNNs
- ❑ DL Frameworks
- ❑ Use Cases

Traditional ML Vs DL

Traditional ML requires manual feature extraction/engineering

Feature extraction for unstructured data is very difficult

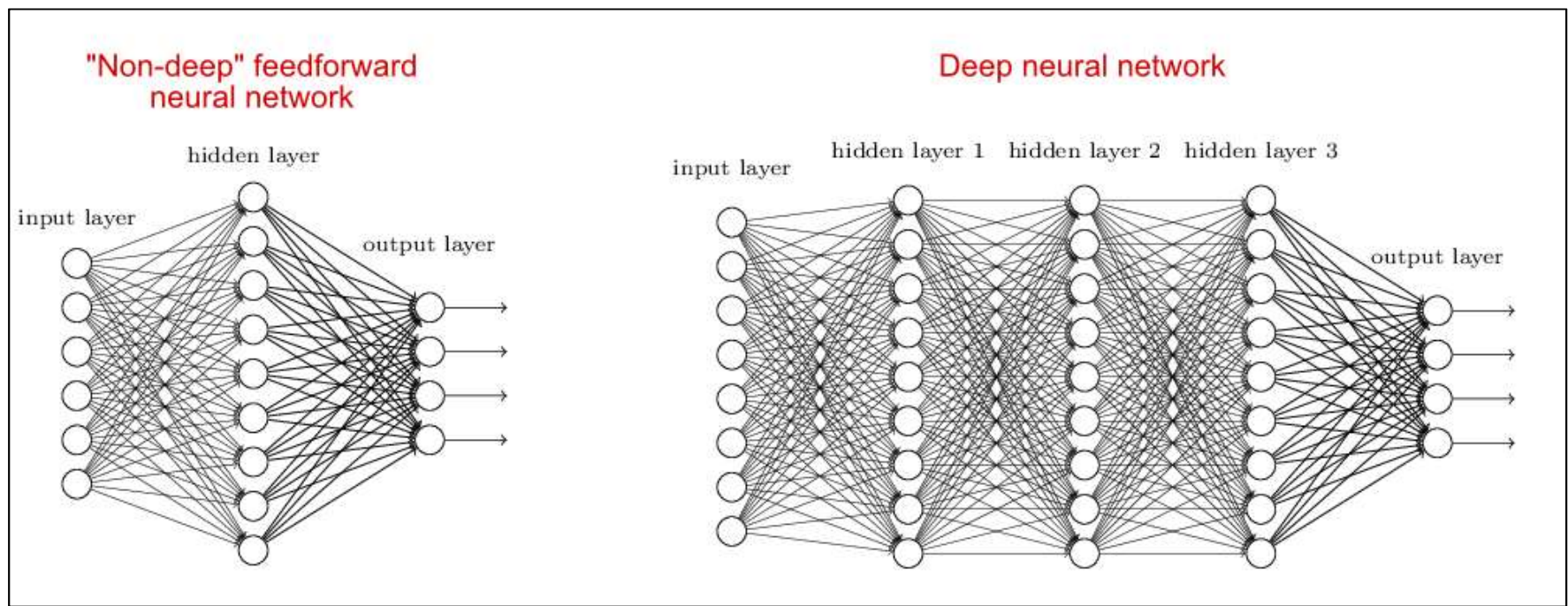
Deep learning can automatically learn features in data

Deep learning is largely a "black box" technique, updating learned weights at each layer

Why is DL popular?

- ❑ DL models has been here for a long time
 - Fukushima (1980) – Neo-Cognitron
 - LeCun (1989) – Convolutional Neural Network
- ❑ DL popularity grew recently
 - With growth of Big Data
 - With the advent of powerful GPUs

Deep Neural Network (DNN)



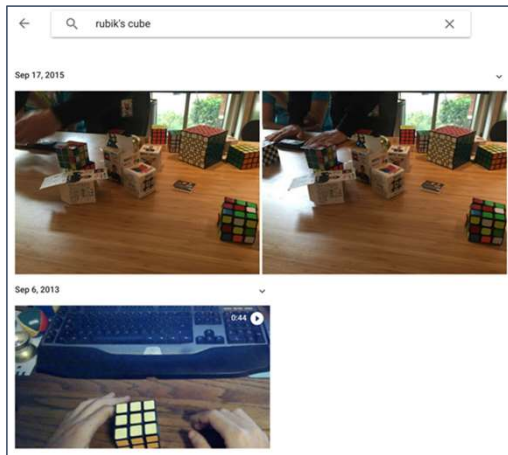
Common DNNs

- ❑ Deep Convolutional Neural Network (DCNN)
 - To extract representation from images

- ❑ Recurrent Neural Network (RNN)
 - To extract representation from sequential data

- ❑ Deep Belief Neural Network (DBN)
 - To extract hierarchical representation from a dataset

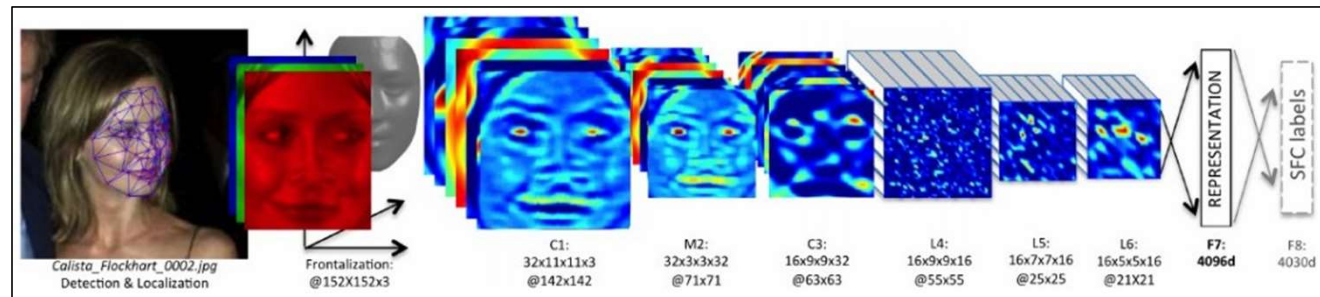
Supervised: ConvNets are everywhere



e.g. Google Photos search



[Goodfellow et al. 2014]



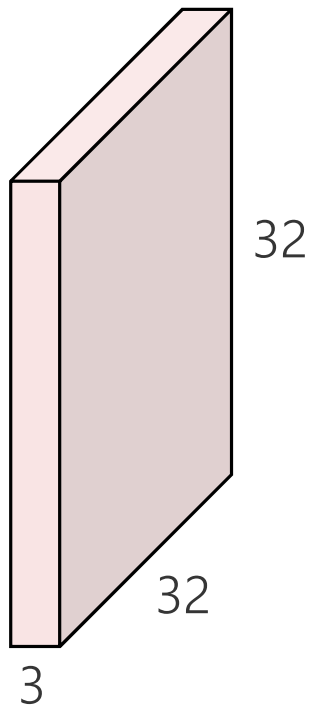
Face Verification, Taigman et al. 2014 (FAIR)



Self-driving cars

Convolution Layer

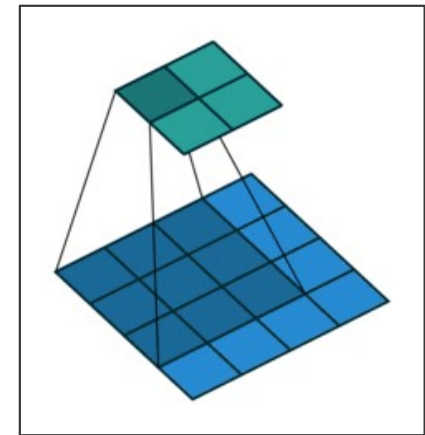
32x32x3 image ← Filters always extend the full depth of the input volume



5x5x3 filter



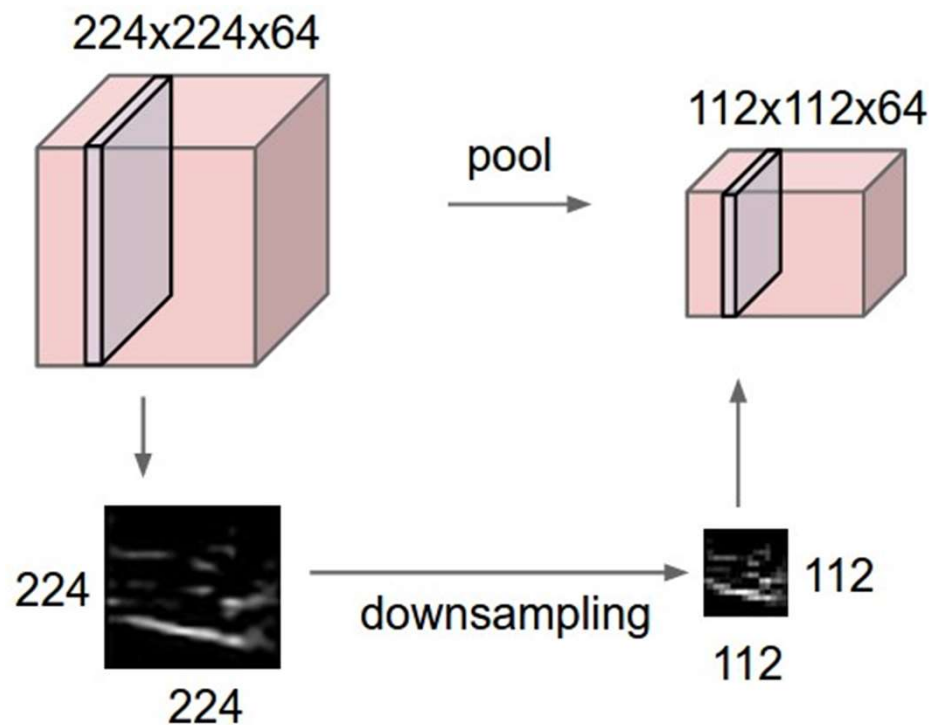
Convolve the filter with the image
i.e. "slide over the image spatially,
computing dot products"



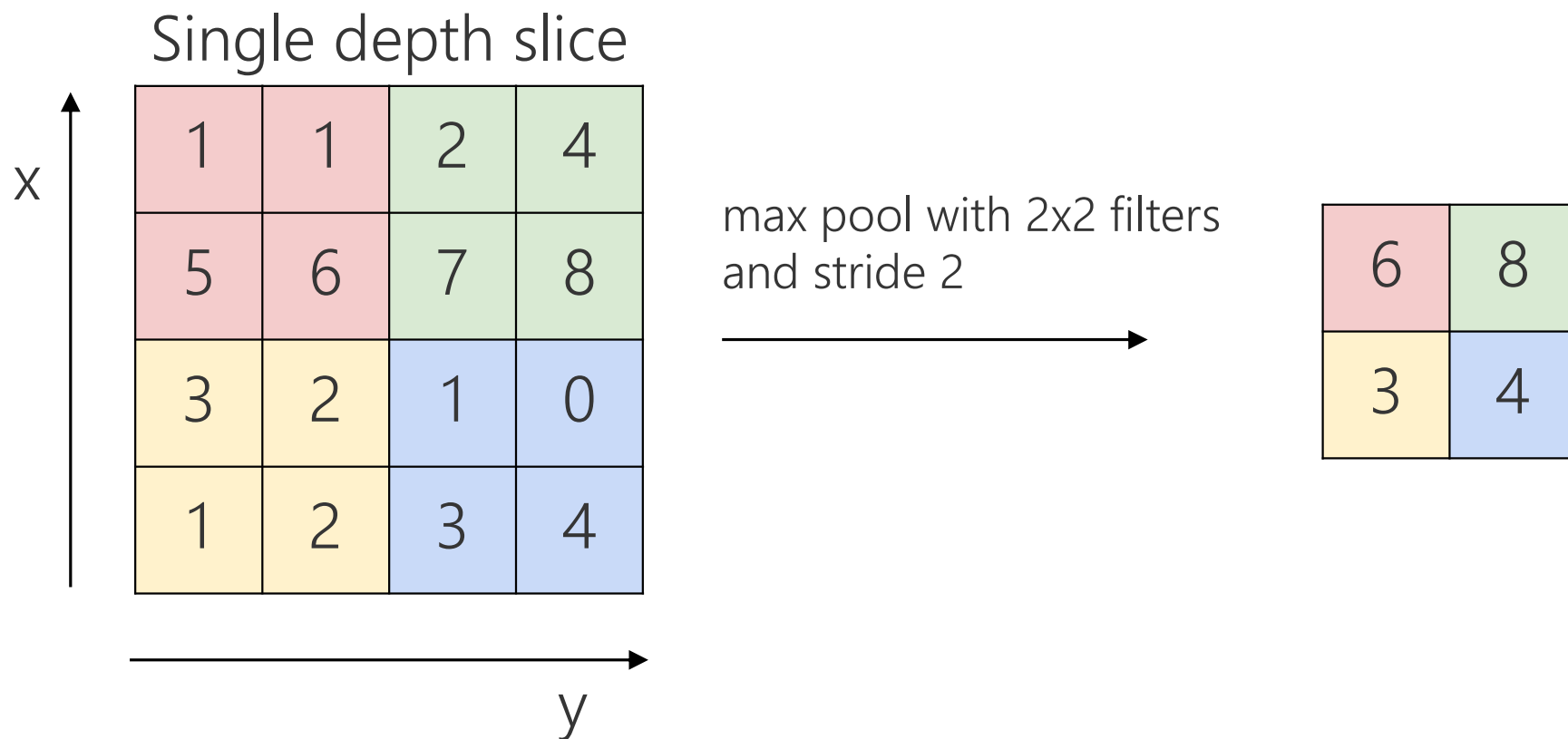
Convoluting a 3 X 3 kernel over
a 4 X 4 input

Pooling layer

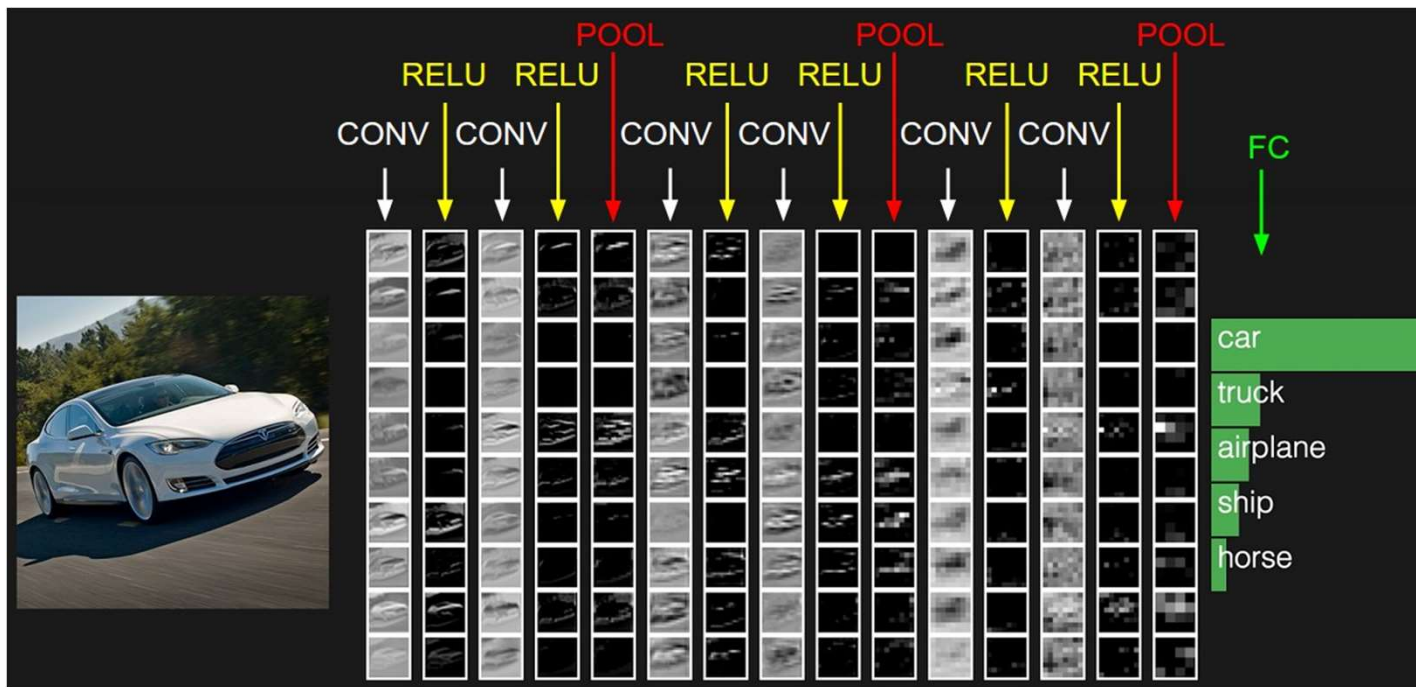
- ❑ makes the representations smaller and more manageable
- ❑ operates over each activation map independently



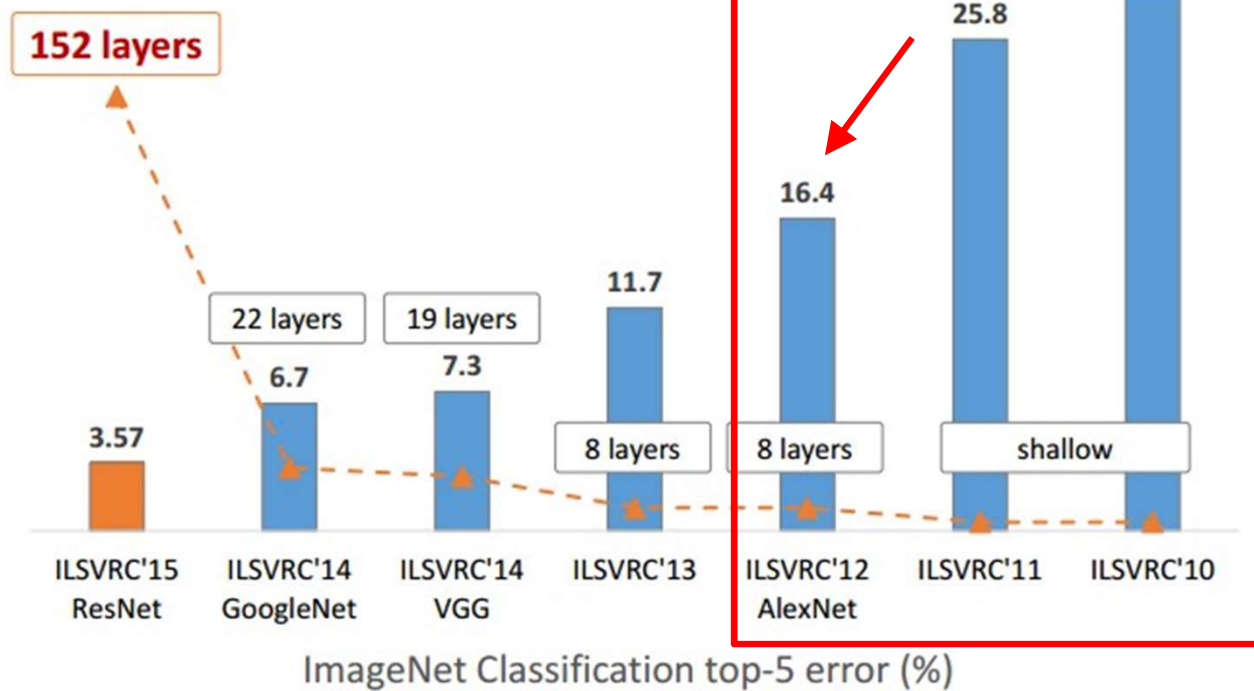
Max Pooling



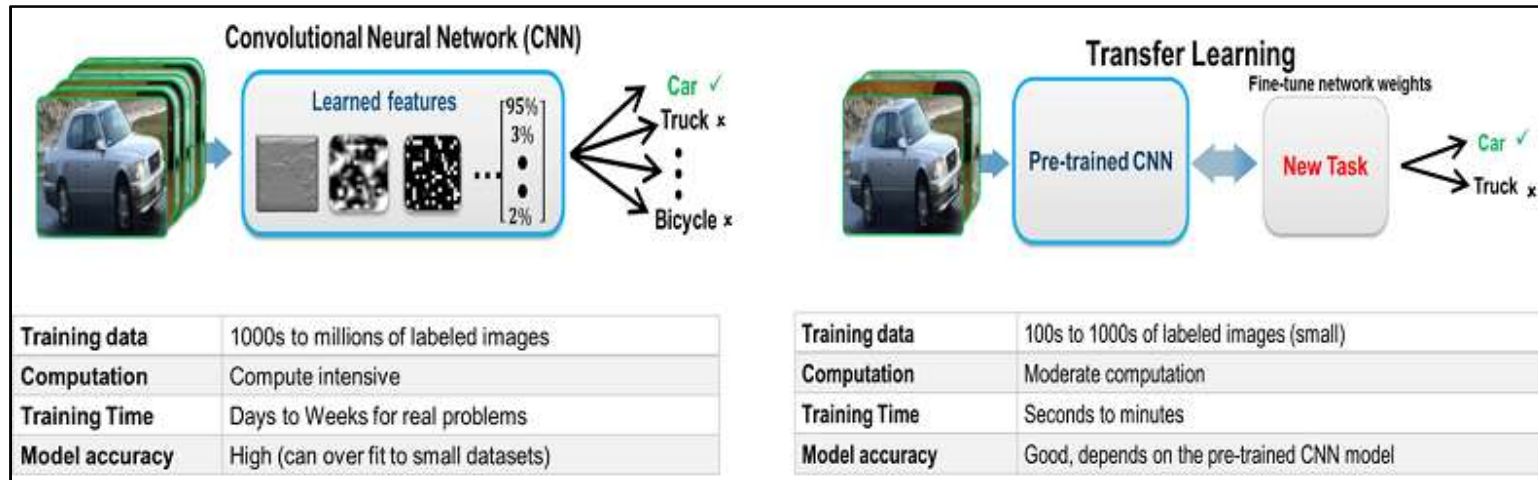
Fully Connected Layer (FC layer)



Revolution of Depth



Transfer Learning & Fine-tuning

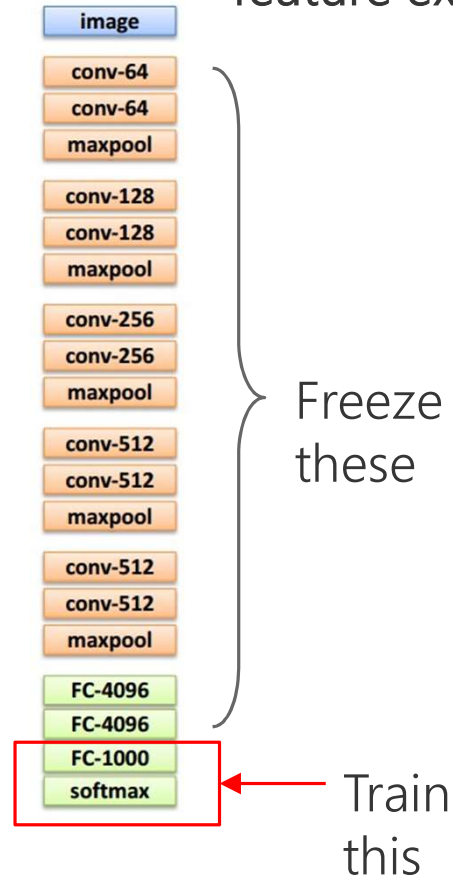


Transfer Learning

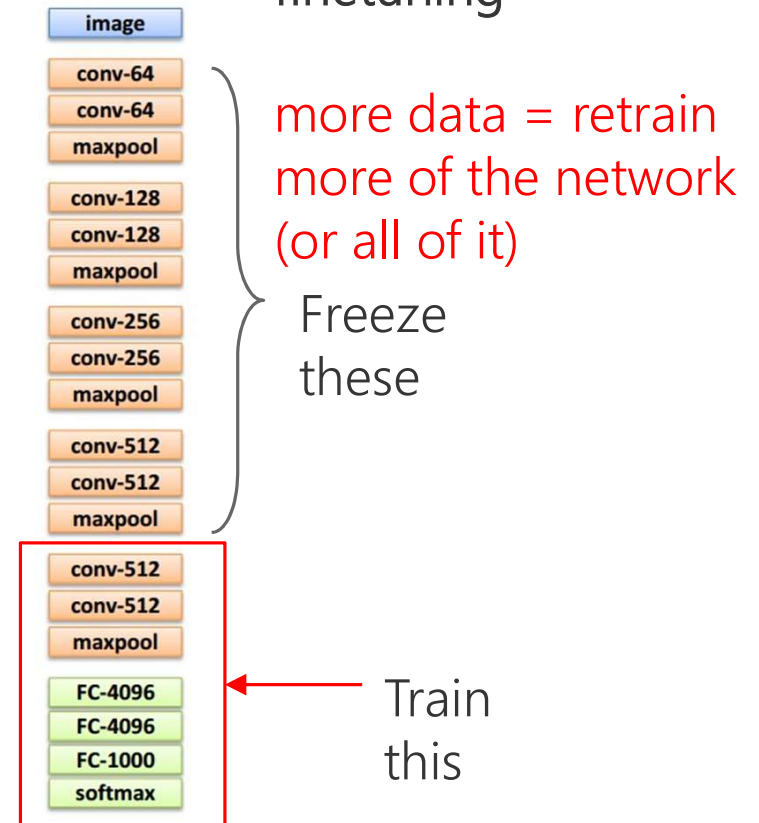
1. Train on Imagenet



2. Small dataset:
feature extractor



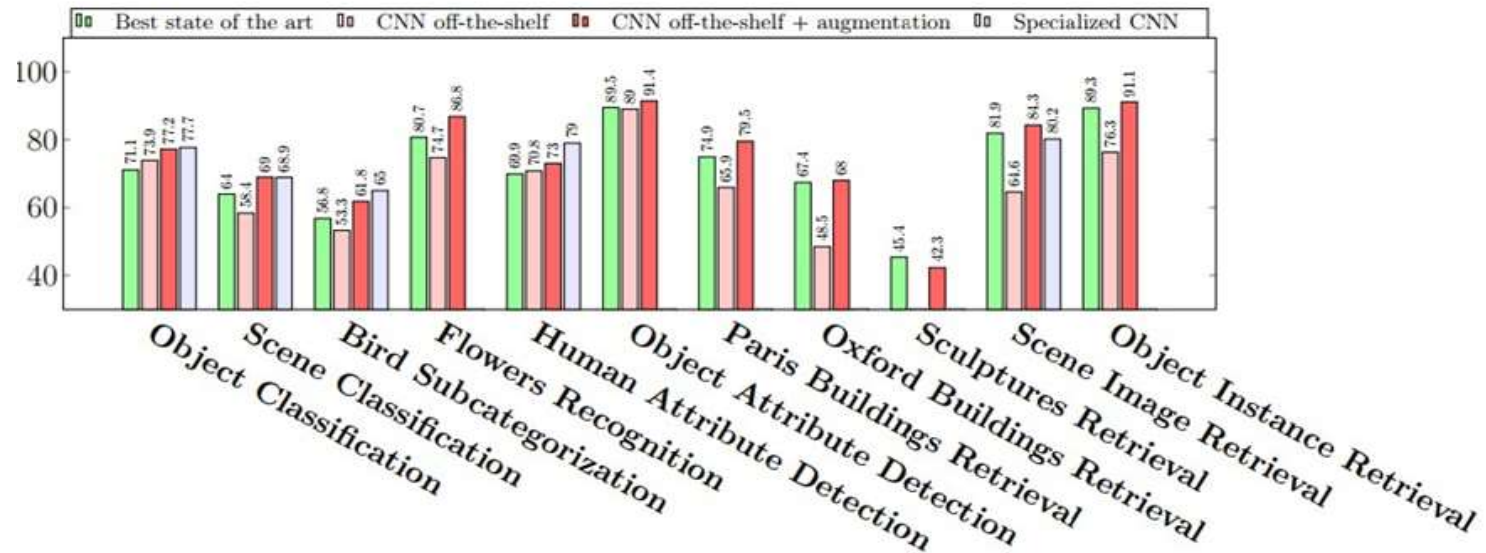
3. Medium dataset:
finetuning



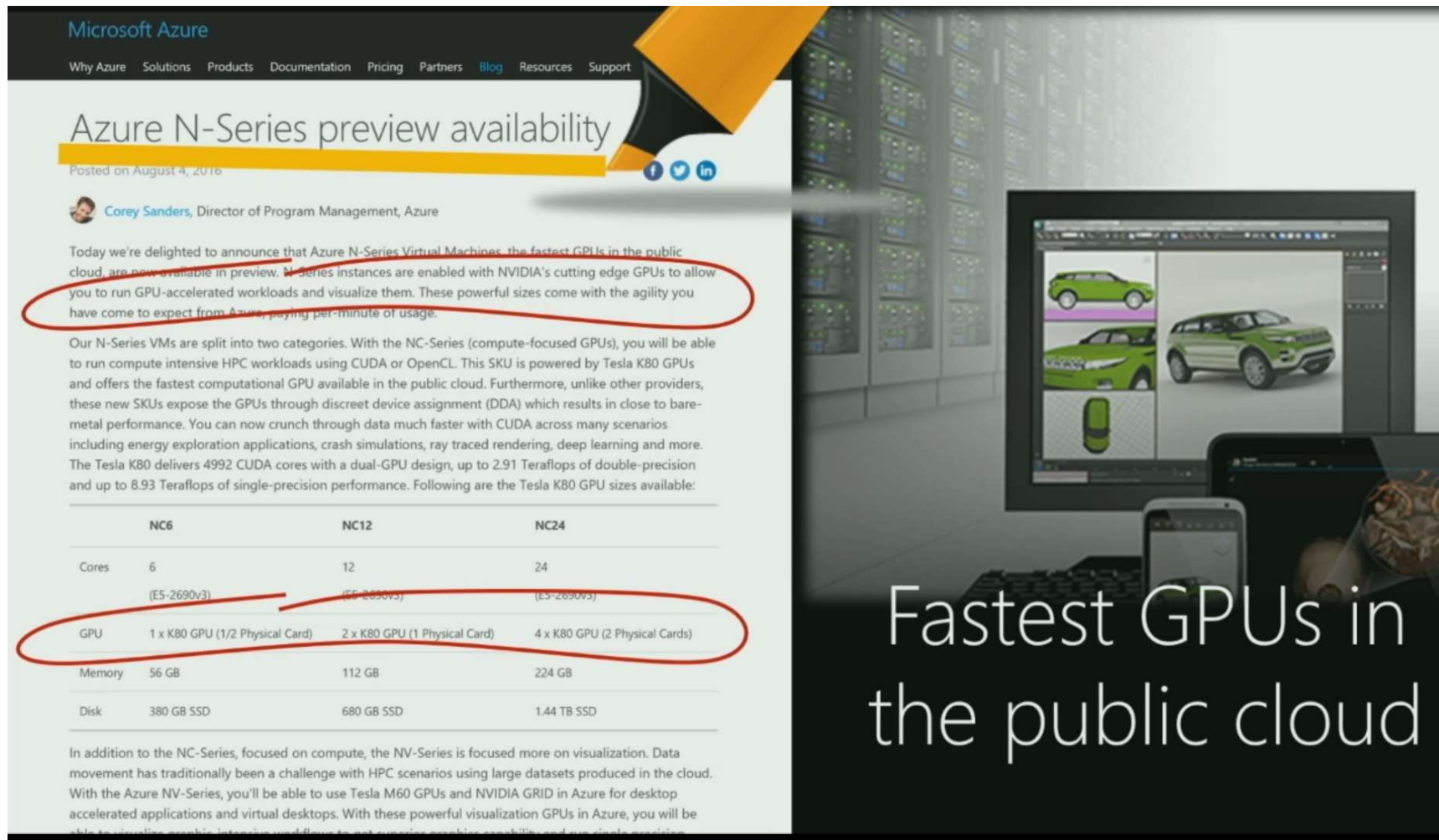
Transfer Learning

CNN Features off-the-shelf: an Astounding Baseline for Recognition

[Razavian et al, 2014]



GPUs in Azure



Microsoft Azure

Why Azure Solutions Products Documentation Pricing Partners Blog Resources Support

Azure N-Series preview availability

Posted on August 4, 2016

Corey Sanders, Director of Program Management, Azure

Today we're delighted to announce that Azure N-Series Virtual Machines, the fastest GPUs in the public cloud, are now available in preview. N-Series instances are enabled with NVIDIA's cutting edge GPUs to allow you to run GPU-accelerated workloads and visualize them. These powerful sizes come with the agility you have come to expect from Azure, paying per-minute of usage.

Our N-Series VMs are split into two categories. With the NC-Series (compute-focused GPUs), you will be able to run compute intensive HPC workloads using CUDA or OpenCL. This SKU is powered by Tesla K80 GPUs and offers the fastest computational GPU available in the public cloud. Furthermore, unlike other providers, these new SKUs expose the GPUs through discreet device assignment (DDA) which results in close to bare-metal performance. You can now crunch through data much faster with CUDA across many scenarios including energy exploration applications, crash simulations, ray traced rendering, deep learning and more. The Tesla K80 delivers 4992 CUDA cores with a dual-GPU design, up to 2.91 Teraflops of double-precision and up to 8.93 Teraflops of single-precision performance. Following are the Tesla K80 GPU sizes available:

	NC6	NC12	NC24
Cores	6 (E5-2690v3)	12 (E5-2690v3)	24 (E5-2690v3)
GPU	1 x K80 GPU (1/2 Physical Card)	2 x K80 GPU (1 Physical Card)	4 x K80 GPU (2 Physical Cards)
Memory	56 GB	112 GB	224 GB
Disk	380 GB SSD	680 GB SSD	1.44 TB SSD

In addition to the NC-Series, focused on compute, the NV-Series is focused more on visualization. Data movement has traditionally been a challenge with HPC scenarios using large datasets produced in the cloud. With the Azure NV-Series, you'll be able to use Tesla M60 GPUs and NVIDIA GRID in Azure for desktop accelerated applications and virtual desktops. With these powerful visualization GPUs in Azure, you will be able to visualize graphic intensive workloads to get answers quicker, simplify and visualize results.

Fastest GPUs in the public cloud

Azure GPU DSVM

- ❑ Used Ubuntu 16.04 N-Series VM
- ❑ NC24 VM with 4 NVIDIA Tesla K80

```
@AzureGPUcluster:~$ nvidia-smi
Mon Sep 26 22:38:34 2016

+-----+
| NVIDIA-SMI 367.18      Driver Version: 367.18 |
+-----+-----+
| GPU  Name            Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf      Pwr:Usage/Cap|      Memory-Usage | GPU-Util  Compute M. |
+-----+-----+
|  0   Tesla K80        Off          | 80F9:00:00.0  Off  | 0          Default |
| N/A   43C    P8       28W / 149W | 0MiB / 11439MiB | 0%               |
+-----+-----+
|  1   Tesla K80        Off          | 9A76:00:00.0  Off  | 0          Default |
| N/A   35C    P8       34W / 149W | 0MiB / 11439MiB | 0%               |
+-----+-----+
|  2   Tesla K80        Off          | 9CC7:00:00.0  Off  | 0          Default |
| N/A   42C    P8       26W / 149W | 0MiB / 11439MiB | 0%               |
+-----+-----+
|  3   Tesla K80        Off          | A5DB:00:00.0  Off  | 0          Default |
| N/A   41C    P0       74W / 149W | 0MiB / 11439MiB | 0%               |
+-----+-----+

+-----+
| Processes: | GPU Memory |
| GPU        PID  Type  Process name      Usage |
+-----+
| No running processes found |
+-----+
```

The Microsoft Cognitive Toolkit (CNTK)

- CNTK expresses (nearly) **arbitrary neural networks** by composing simple building blocks into complex **computational networks**, supporting relevant network types and applications.
- CNTK is **production-ready**: State-of-the-art accuracy, efficient, and scales to multi-GPU/multi-server.

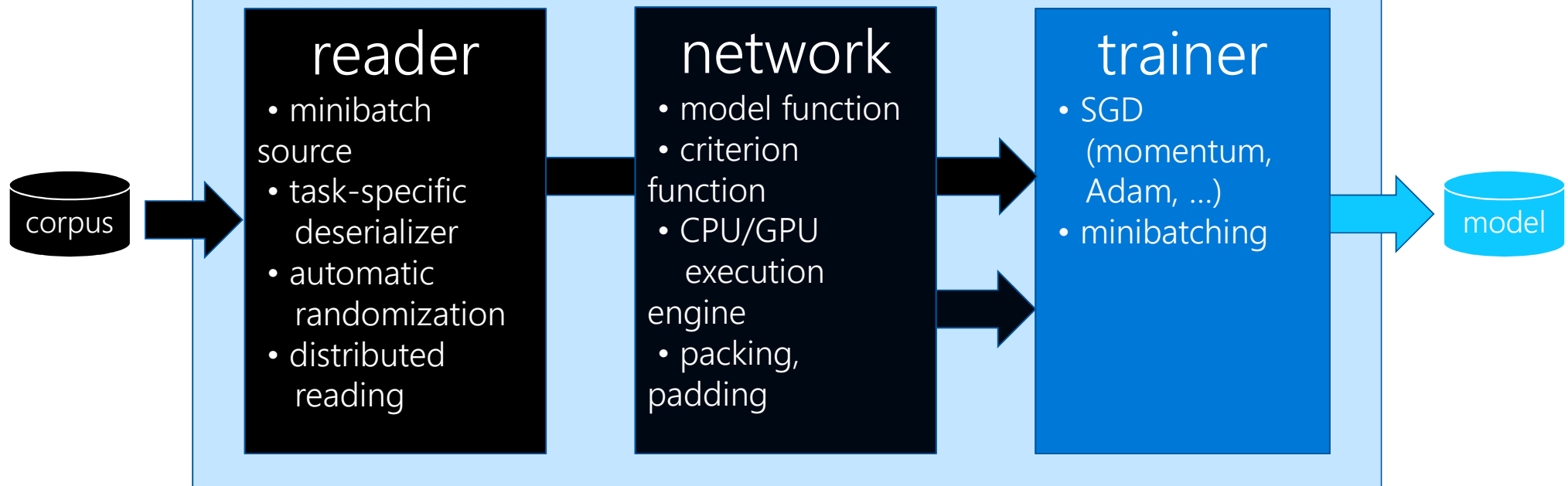


Microsoft Cognitive Toolkit

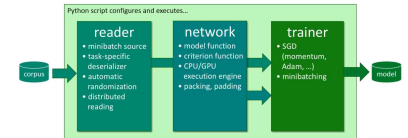
- ease of use
 - *what*, not *how*
 - powerful stock library
- fast
 - optimized for NVidia GPUs & libraries
 - best-in-class multi-GPU/multi-server algorithms
- flexible
 - powerful & composable Python and C++ API
- Linux and Windows
- Development is completely in the open

Anatomy of a training job

Python script configures and executes...



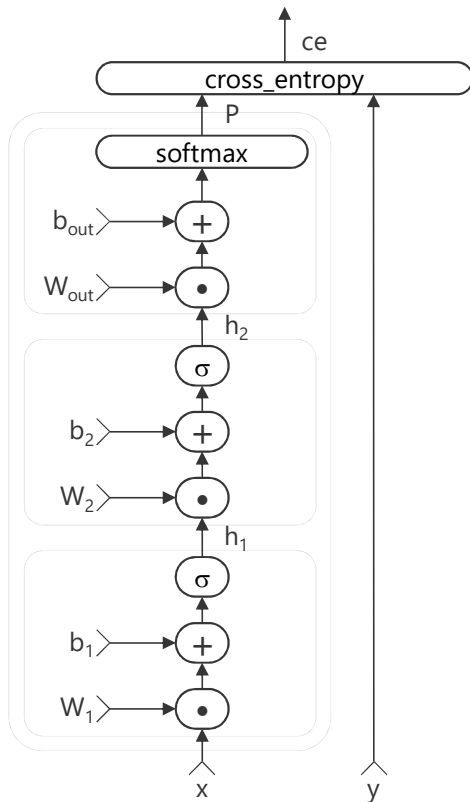
how to: reader



```
def create_reader(map_file, mean_file, is_training):
    # image preprocessing pipeline
    transforms = [
        ImageDeserializer.crop(crop_type='Random', ratio=0.8, jitter_type='uniRatio')
        ImageDeserializer.scale(width=image_width, height=image_height, channels=num_channels,
                                interpolations='linear'),
        ImageDeserializer.mean(mean_file)
    ]
    # deserializer
    return MinibatchSource(ImageDeserializer(map_file, StreamDefs(
        features = StreamDef(field='image', transforms=transforms), '
        labels   = StreamDef(field='label', shape=num_classes)
    )), randomize=is_training, epoch_size = INFINITELY_REPEAT if is_training else FULL_DATA_SWEEP)
```

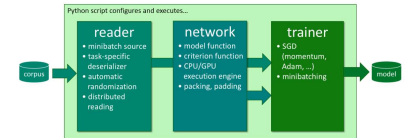
- automatic on-the-fly randomization important for large data sets
- readers compose, e.g. image → text caption

how to: network

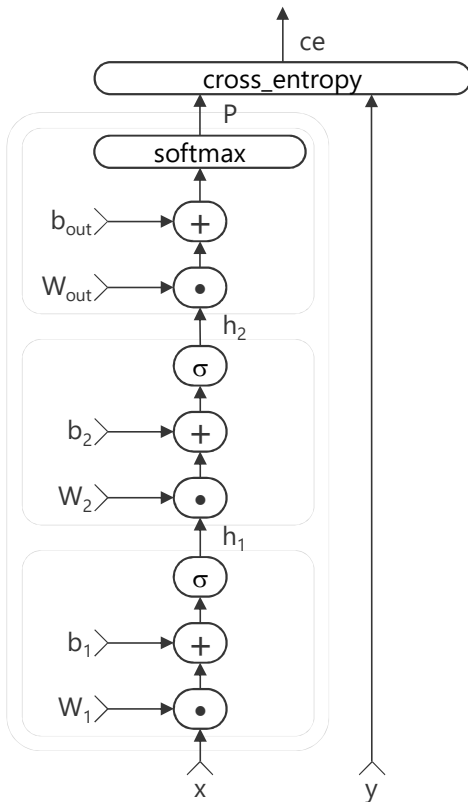
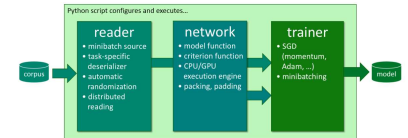


```
M = 40 ; N = 512 ; J = 9000 # feat/hid/out dim
x = Input(M) ; y = Input(J) # feat/labels
```

```
from layers import *
model = Sequential ([
    Dense(N), sigmoid,
    Dense(N), sigmoid,
    Dense(J), softmax
])
P = model(x)
ce = cross_entropy(P, y)
```



how to: network



`M = 40 ; N = 512 ; J = 9000 ; L = 2`

`x = Input(M) ; y = Input(J) # feat/labels`

`from layers import *`

`model = Sequential ([`

`[Dense(N, activation=sigmoid)`

`for i in range(L)],`

`Dense(J, activation=softmax)`

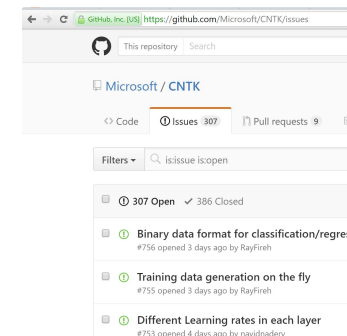
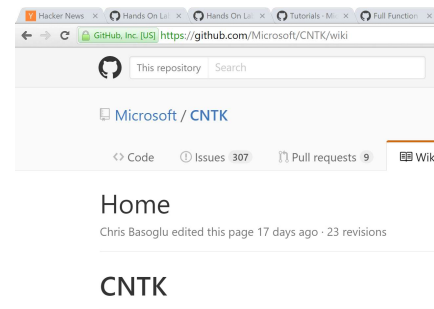
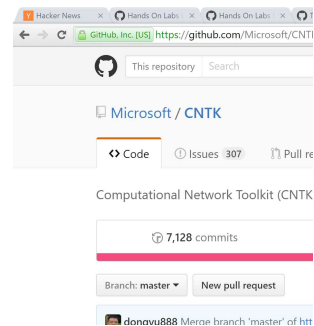
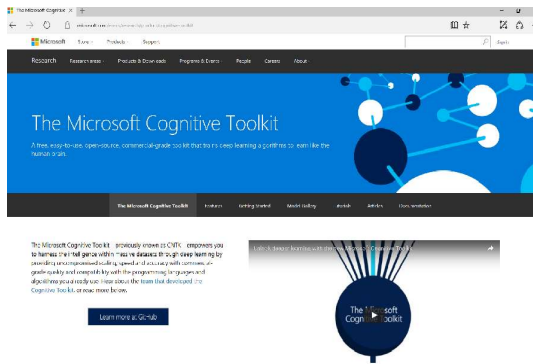
`])`

`P = model(x)`

`ce = cross_entropy(P, y)`

Microsoft Cognitive Toolkit

- Github: <https://github.com/Microsoft/CNTK>
- Wiki: <https://github.com/Microsoft/CNTK/wiki>
- Issues: <https://github.com/Microsoft/CNTK/issues>
- cntkhelp@microsoft.com



Hands-on

