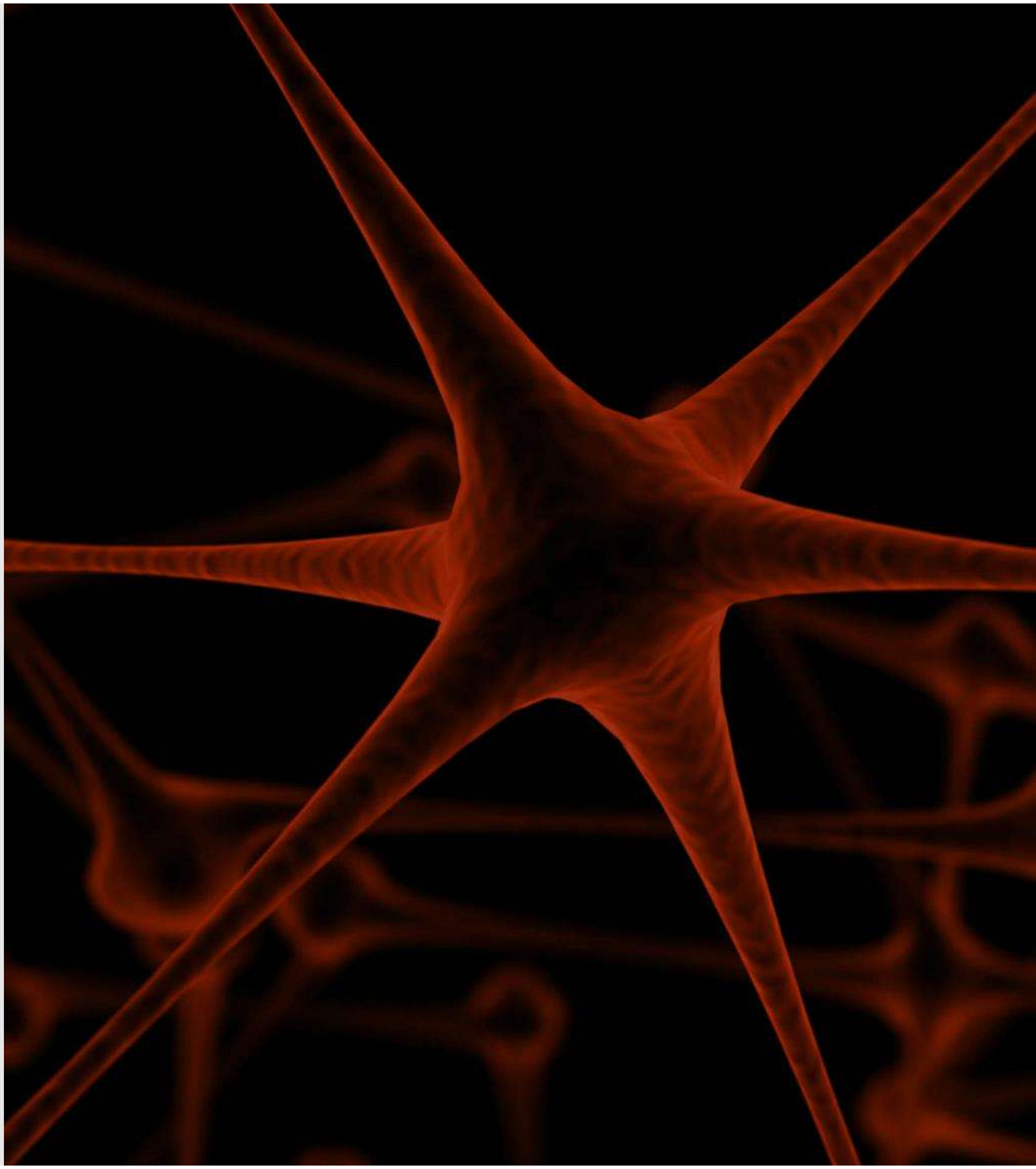
Microsoft

De-mystifying Deep Learning

Anusua Trivedi
Data Scientist
Email: antriv@microsoft.com
Twitter: @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

Deep learning can automatically learn features in data

Feature extraction for unstructured data is very difficult

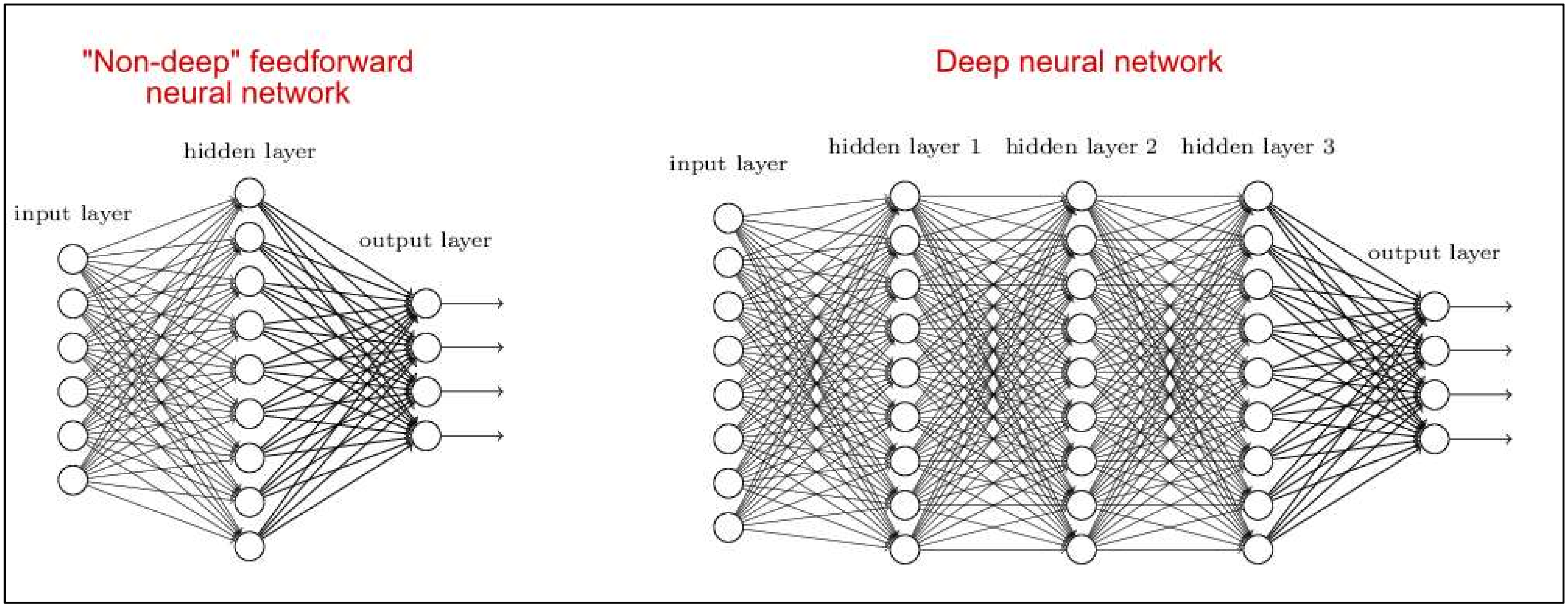
Deep learning is largely a "black box" technique, updating learned weights at each layer

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)



*http://neuralnetworksanddeeplearning.com/chap5.html

Common DNNs

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

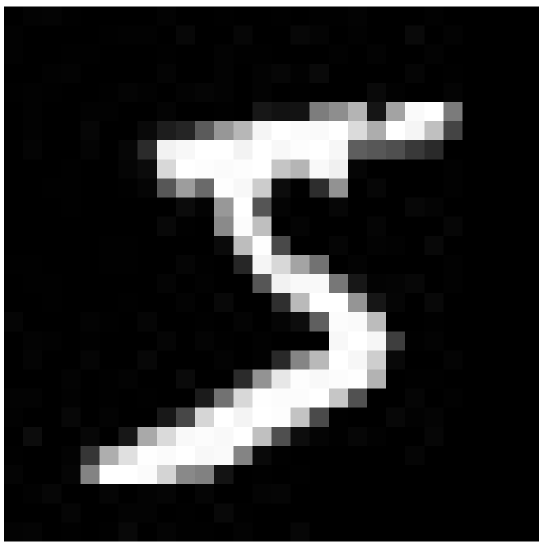
- Recurrent Neural Network (RNN)
  - To extract representation from sequential data

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

Deep learning and computer vision

Vision is hard

Vision is hard because images are big matrices of numbers.



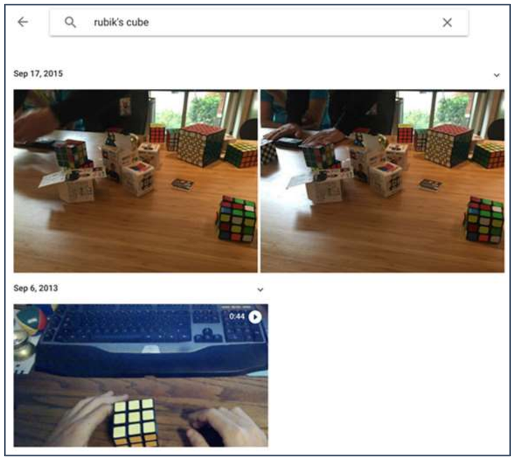
Example from MNIST handwritten digit dataset [LeCun and Cortes, 1998].

How a computer sees an image

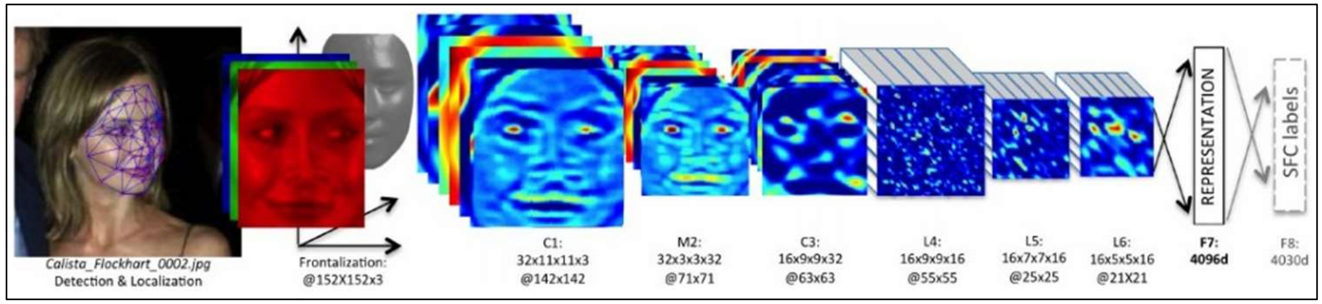
[22, 81, 44, 88, 17, 0, ..., 45]

- Even harder for 3D objects.
- You move a bit, and everything changes.

Supervised: ConvNets are everywhere



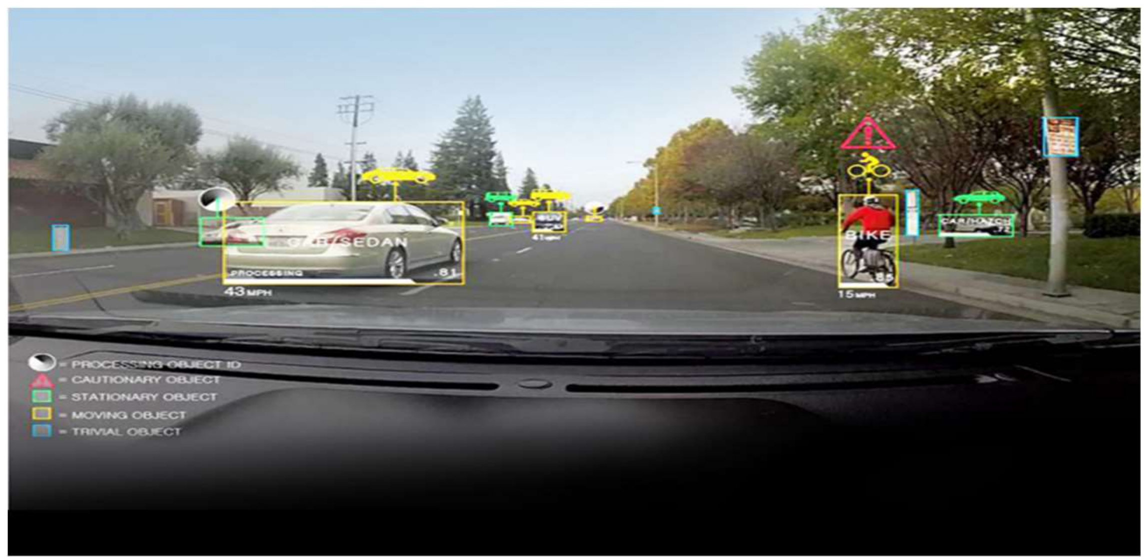
e.g. Google Photos search



Face Verification, Taigman et al. 2014 (FAIR)

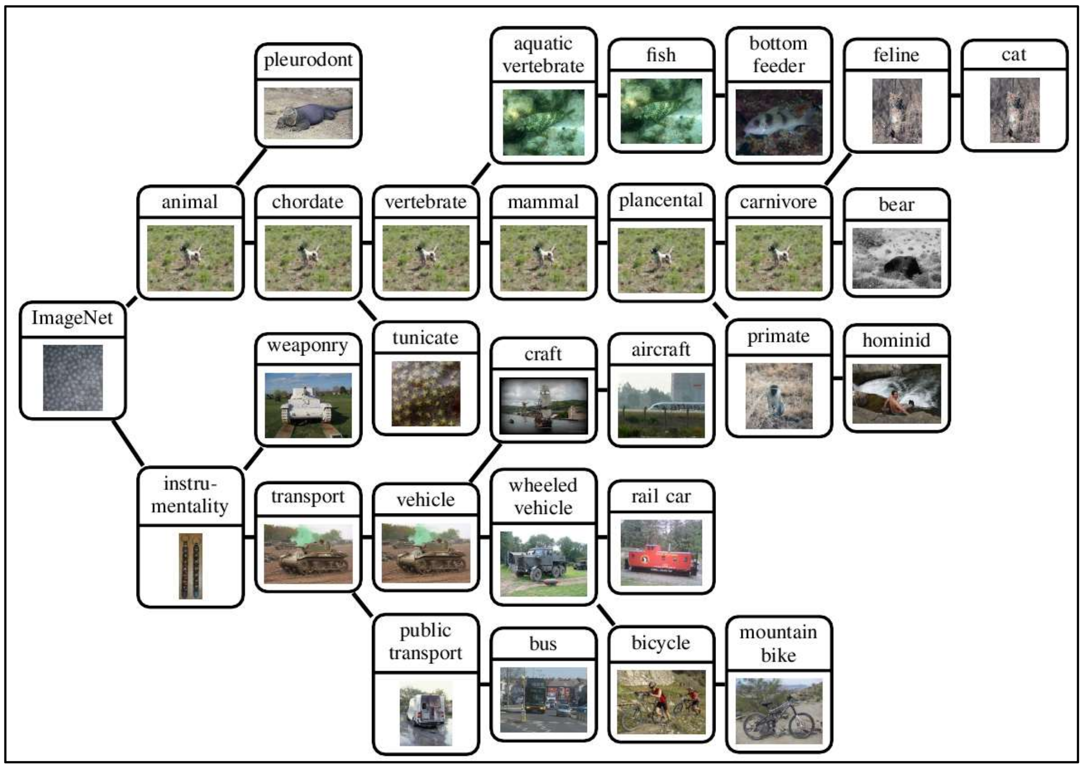
[Goodfellow et al. 2014]

*Andrej Karpathy's recent presentation



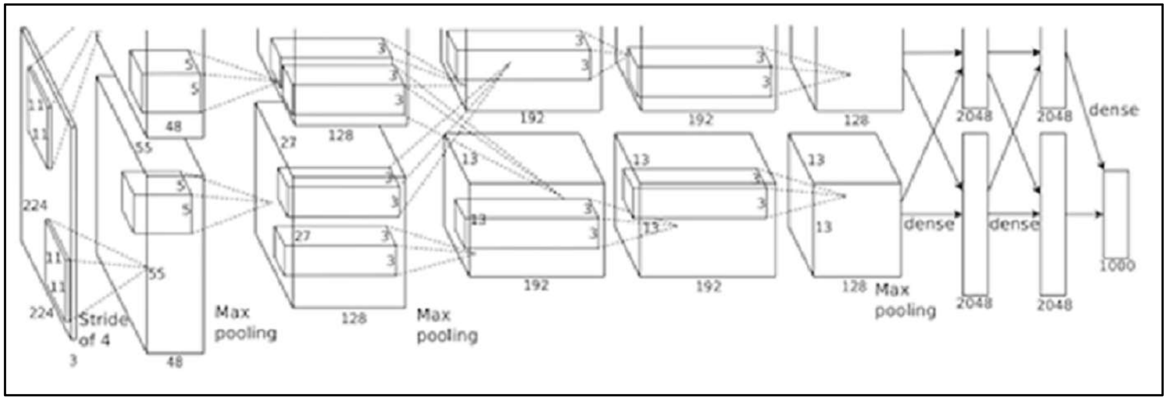
Self-driving cars

IMAGENET



*http://groups.inf.ed.ac.uk/calvin/imagenet/prototypes.html

AlexNet



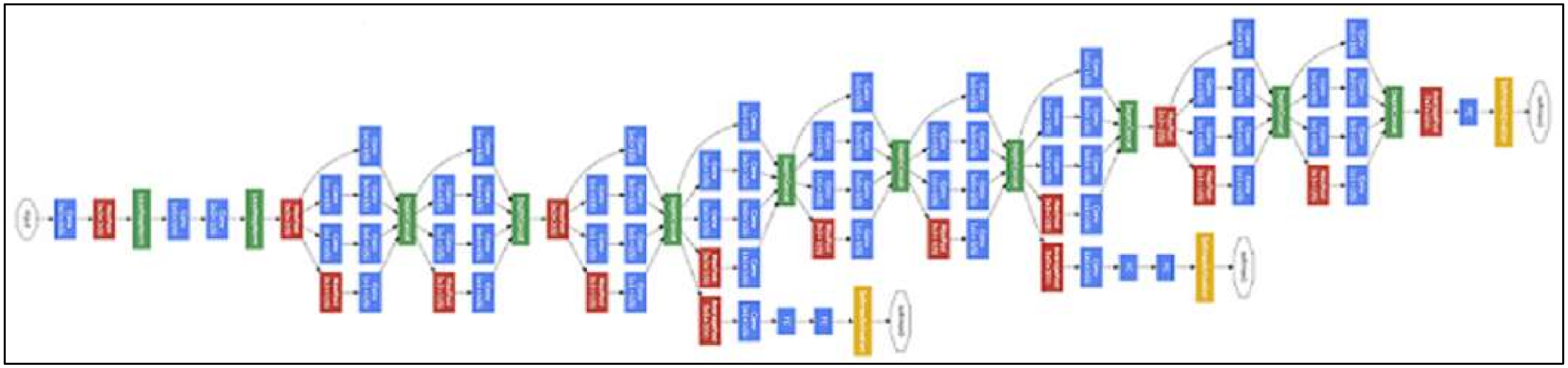
*https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

VGGNet

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

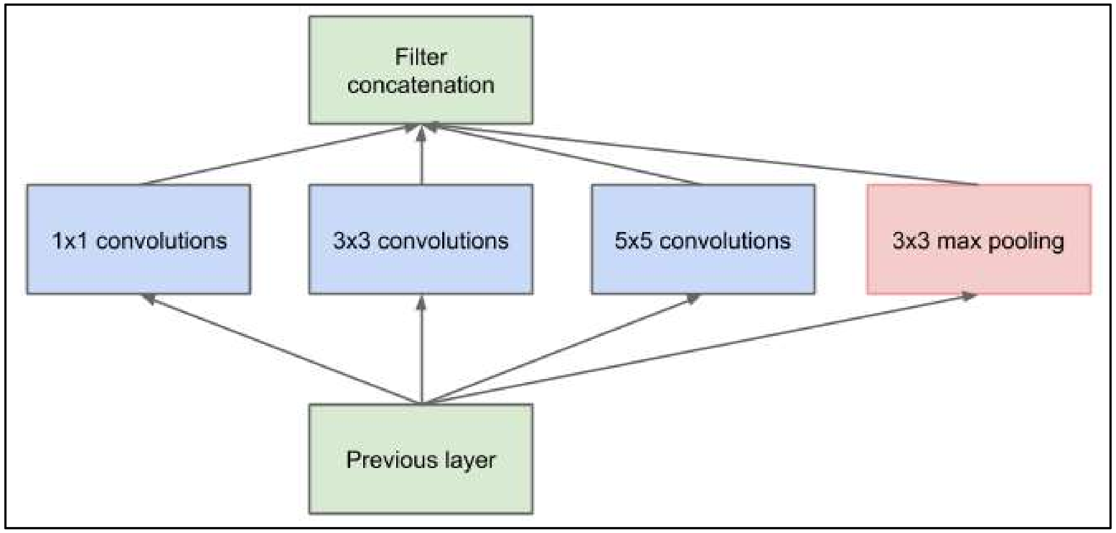
*http://arxiv.org/pdf/1409.1556v6.pdf

GoogLeNet



*http://www.csc.kth.se/~roelof/deepdream/bvlc_gnet.html

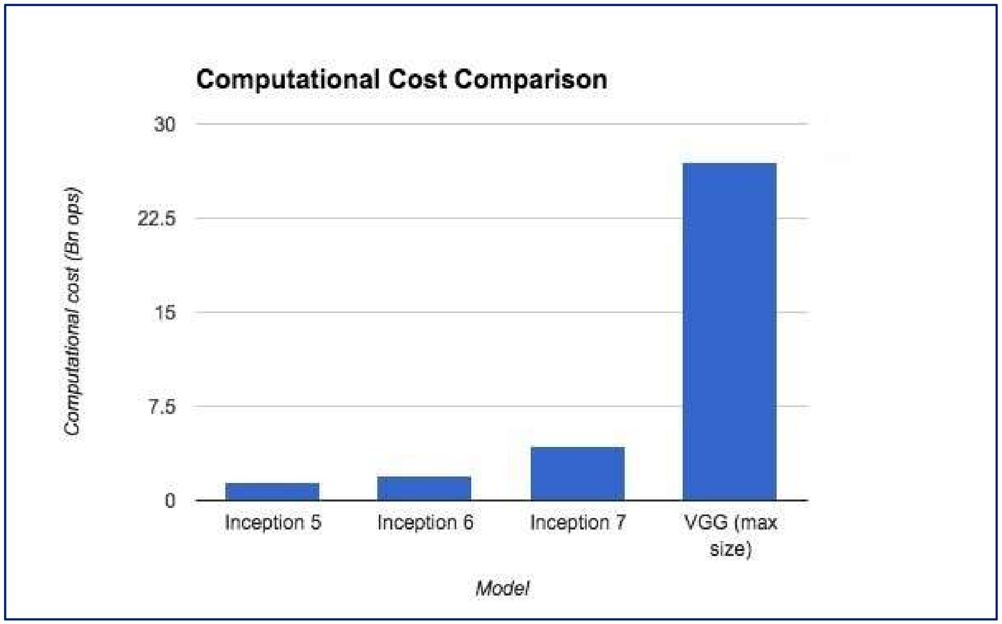
GoogLeNet uses Inception



Inception Module

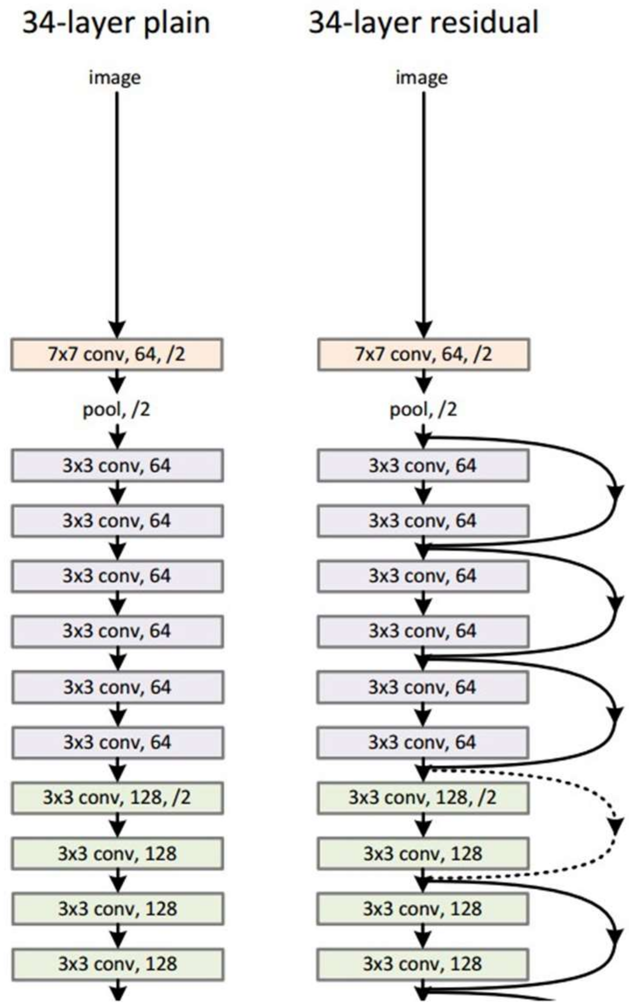
*https://arxiv.org/pdf/1409.4842v1.pdf

Inception Performance comparison

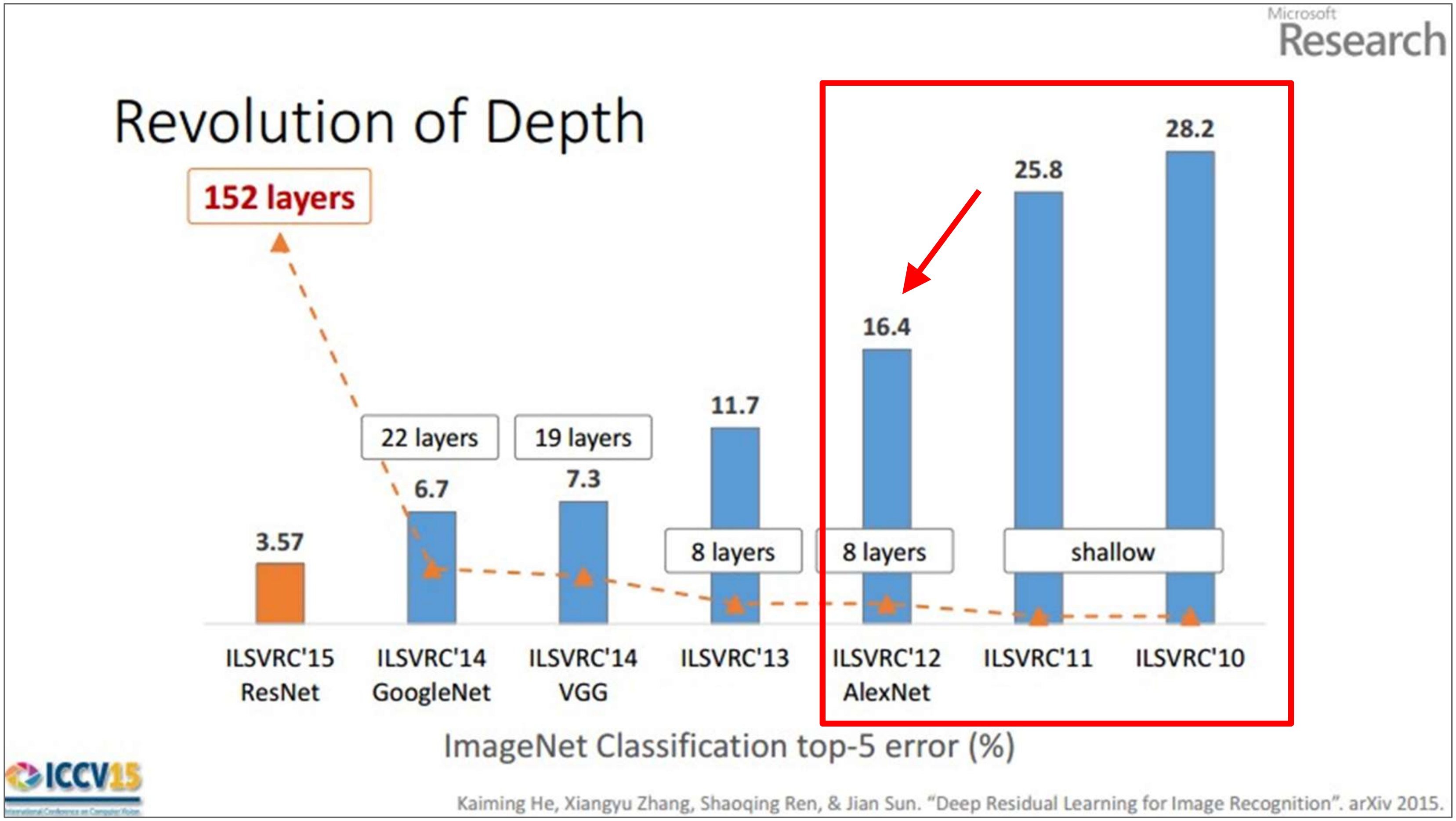


*http://lsun.cs.princeton.edu/slides/Christian.pdf

Microsoft ResNet



*https://arxiv.org/pdf/1512.03385v1.pdf



*Andrej Karpathy's recent presentation

Deep learning and natural language processing

Deep learning enables sub-symbolic processing

I	<i>
bought	<bought>
a	<a>
car	<car>
.	<.>

You have to remember to represent “purchased” and “automobile.”

What about “truck”?

How do you encode the meaning of the entire sentence?

But what about a sentence?

Algorithm for generating vectors for sentences

1. Make the sentence vector be the vector for the first word.
2. For each subsequent word, combine its vector with the sentence vector.
3. The resulting vector after the last word is the sentence vector.

Can be implemented using a recurrent neural network (RNN)

Deep learning and question answering

Bob went home.
Tim went to the junkyard.
Bob picked up the jar.
Bob went to town.
Where is the jar? A: town

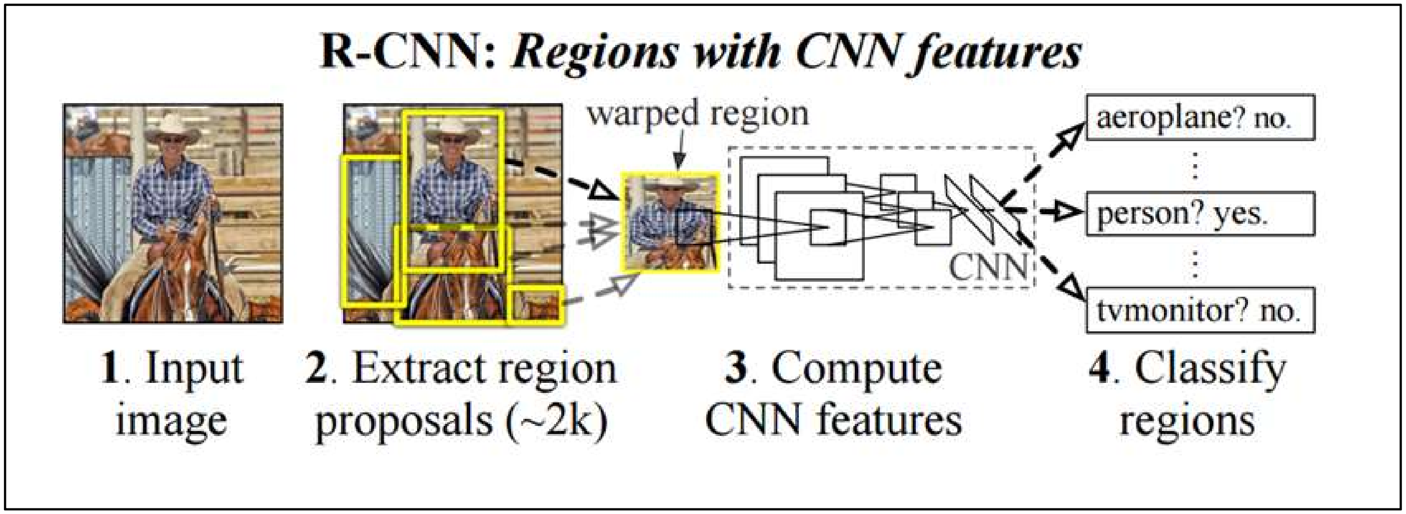
Memory Networks [Weston et al., 2014]: Updates memory vectors based on a question and finds the best one to give the output.

The office is north of the yard.
The bath is north of the office.
The yard is west of the kitchen.
How do you go from the office to the kitchen? A: south, east

Neural Reasoner [Peng et al., 2015]: Encodes the question and facts in many layers, and the final layer is put through a function that gives the answer.

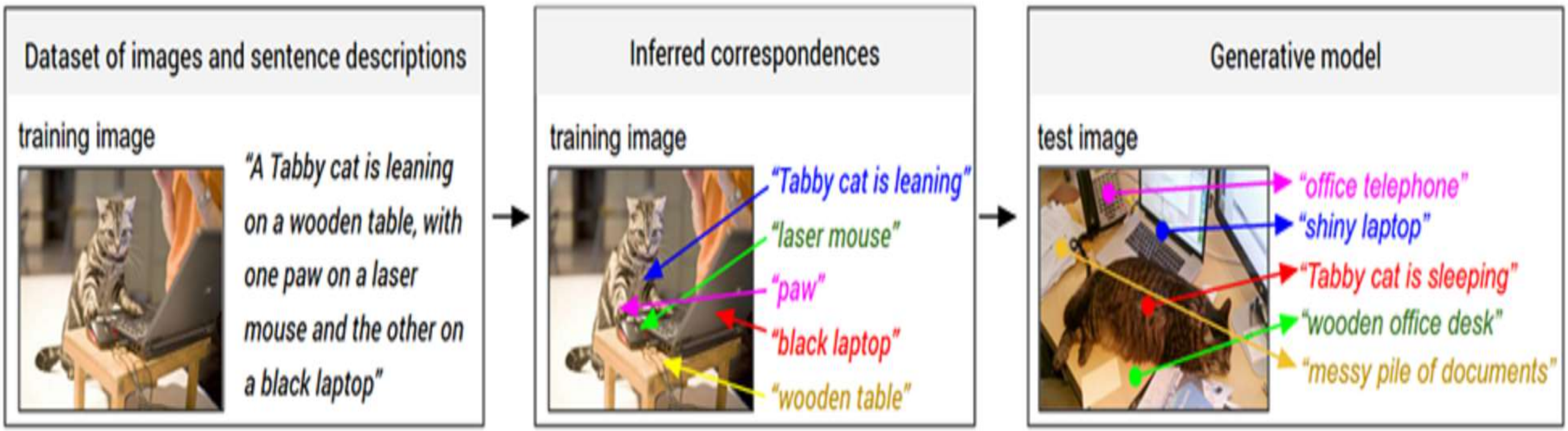
Other commonly used DNNs

Region Based CNN (RCNN)



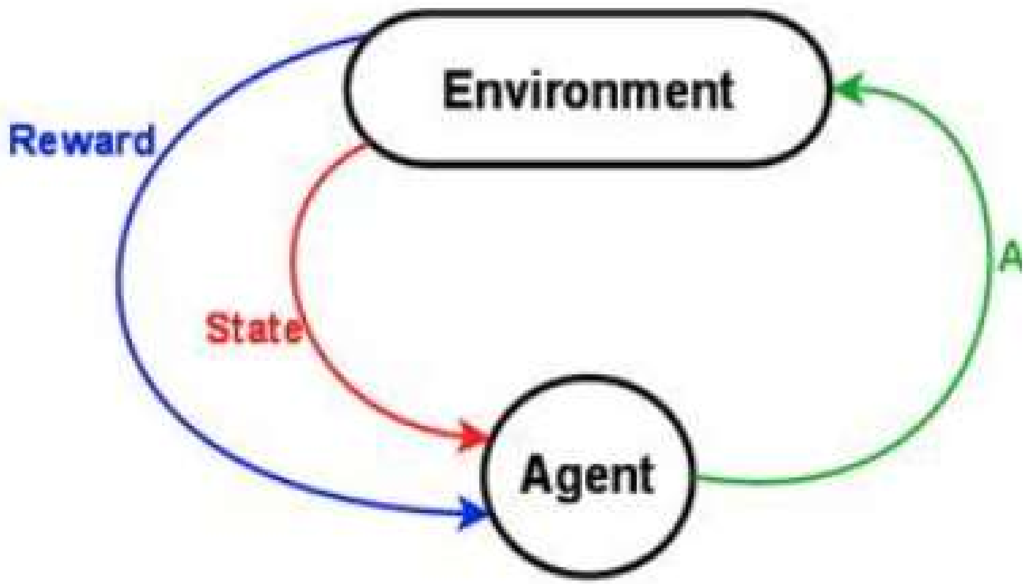
*https://arxiv.org/pdf/1311.2524v5.pdf

Generating Image Descriptions (CNN-RNN)



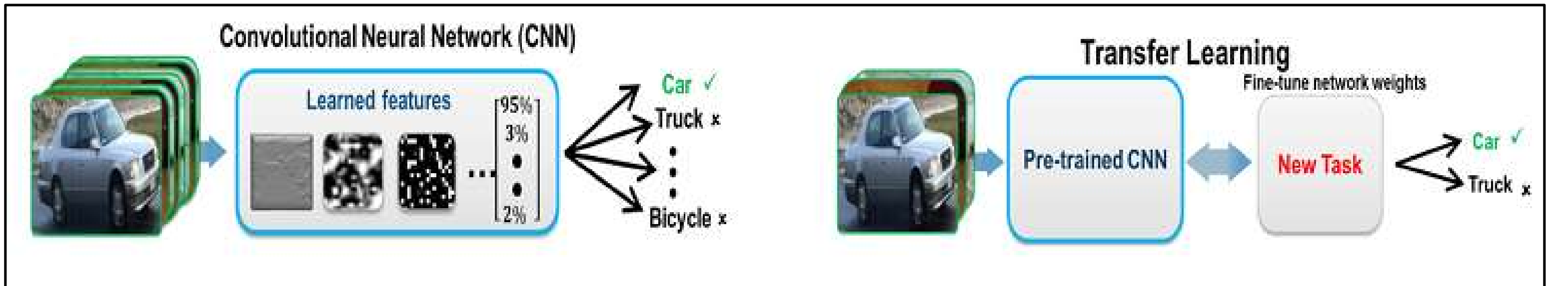
*https://arxiv.org/pdf/1412.2306v2.pdf

Deep Reinforcement Learning



Increasing Re-usability of Deep Learning models

Transfer Learning & Fine-tuning

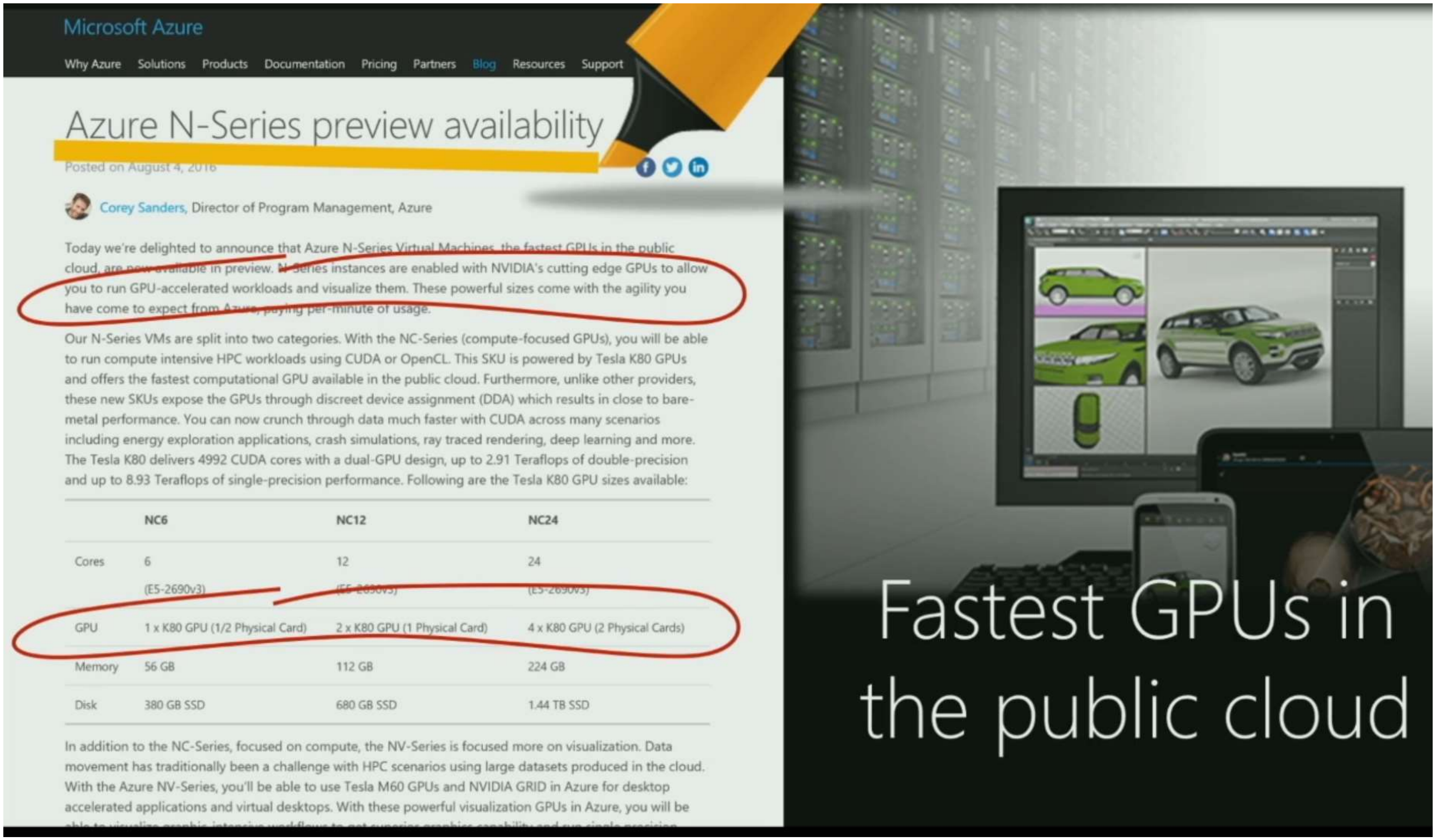


Training data	1000s to millions of labeled images
Computation	Compute intensive
Training Time	Days to Weeks for real problems
Model accuracy	High (can over fit to small datasets)

Training data	100s to 1000s of labeled images (small)
Computation	Moderate computation
Training Time	Seconds to minutes
Model accuracy	Good, depends on the pre-trained CNN model

*http://www.mathworks.com/discovery/deep-learning.html?requestedDomain=uk.mathworks.com

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