Can HPC Technologies Lead the way for AI?

Adnan Khaleel

Global Sales Strategy for HPC & AI

Adnan.Khaleel@dell.com

Sept 2018

AI customers demonstrate the value

MasterCard

Turn 160M transactions/ hour with 1.9M rules to protect against fraud

Caterpillar

Autonomous mining for safety

Zenuity

4.4Pb of data/mo
~50 simulations/ hr
safe autonomous driving

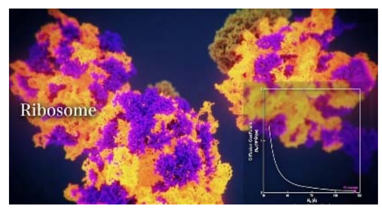
MIT Lincoln Labs

1+ petaFLOP to enable robotic vehicles, cyber security, bioinformatics

Develop, test and run innovative algorithms

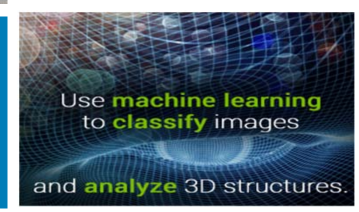
ZIFF.ai

AI startup accelerates training models with 10s of millions of images



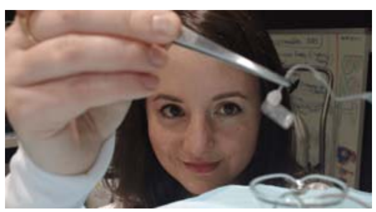
Simon Fraser University

Analyze DNA of microbes and stop outbreaks



CSIRO

More than 1,800 patents in science and technology



AeroFarms

390x more productive than traditional field farming

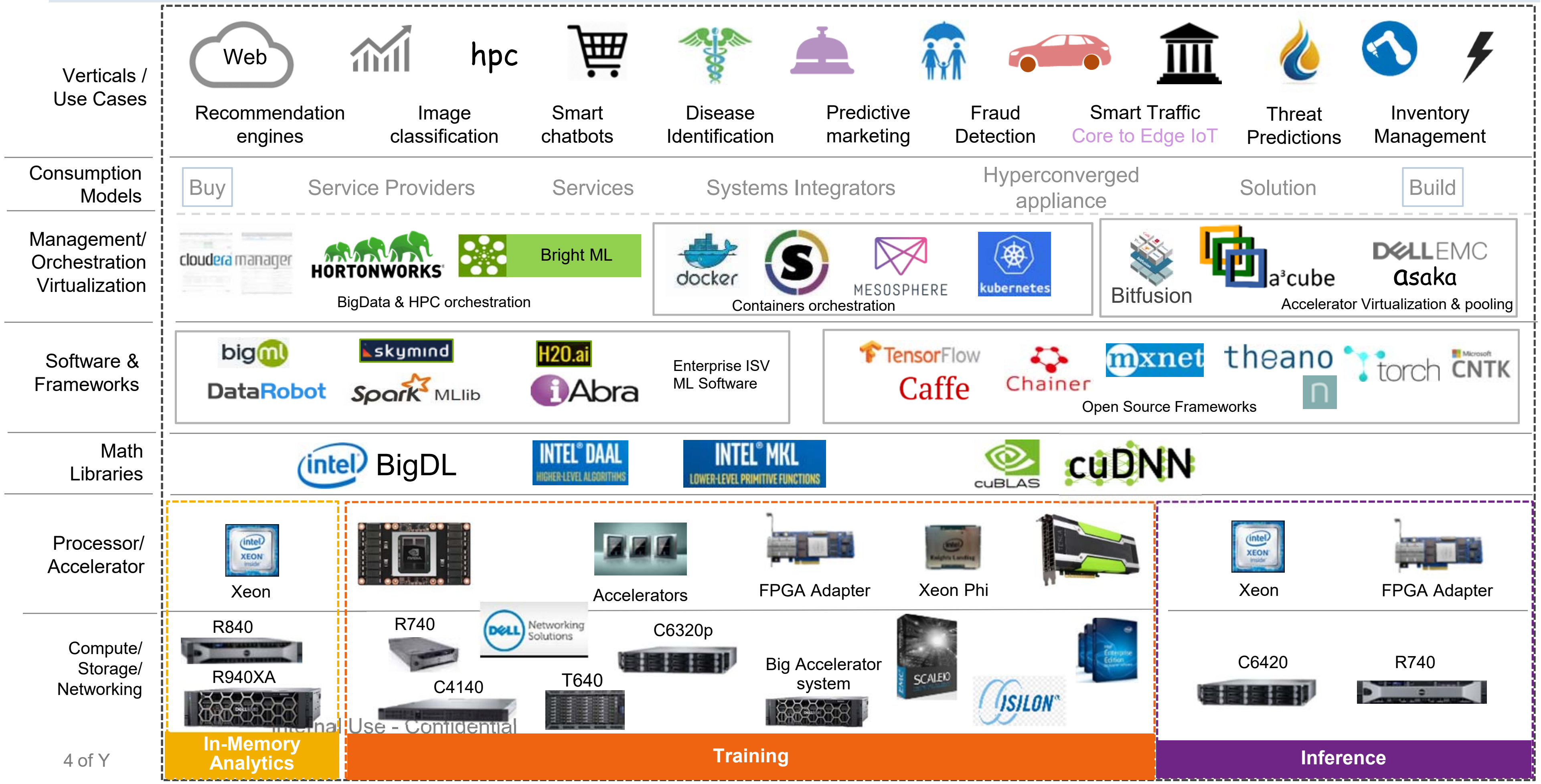
CU

300% faster performance

Motivations for this discussion

- AI is still in its early days as computational technology; lots of dynamism
- Advise CIOs/CTOs and our PoV of how we see the technology evolving
  - Also from a future product planning perspective
  - What can we make an educated guess
- Maximize ROI on current investments
- What will be the right way to “scale” this problem size and or speed?
- It’s easier to come up with an innovative HW architecture than to “actually” program it
- What can we apply from HPC lessons learnt to this domain?
  - Interconnects, memory, scale-out within nodes vs across nodes
    - Are nodes wtih 8, 16, 32 GPUs useful and how far do we go?

Machine Learning & Deep Learning eco-system – solving real world problems



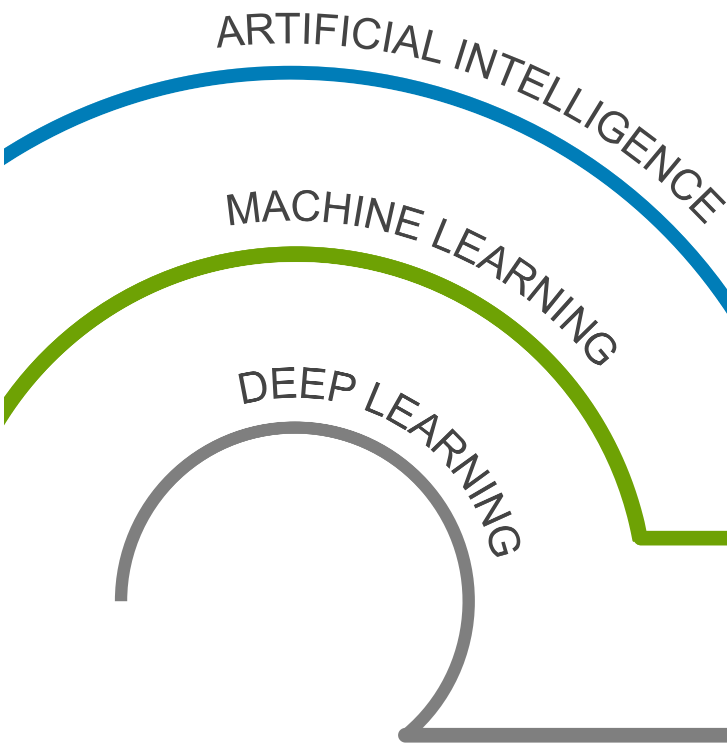
Dell EMC Ready Solutions for Machine & Deep Learning – Clearly Scale-out

Deep Learning with NVIDIA

Deep Learning with Intel

Machine Learning with Hadoop

What is AI, machine learning and deep learning?



AI refers to machine intelligence, while machine learning and deep learning are the technologies that underpin AI and make it possible.

Machine learning refers to the process of “training” the machine, feeding large amounts of data into algorithms that give it the ability to learn how to perform a task.

Deep learning is a machine learning technique that uses neural networks as the underlying architecture for training models.

Quick Primer

Machine Learning

How do you engineer the best features?

$N \times N$

$(f_1, f_2, \ldots, f_K)$

Roundness of face
Dist between eyes
Nose width
Eye socket depth
Cheek bone structure
Jaw line length
…etc.

CLASSIFIER ALGORITHM

SVM
Random Forest
Naïve Bayes
Decision Trees
Logistic Regression
Ensemble methods

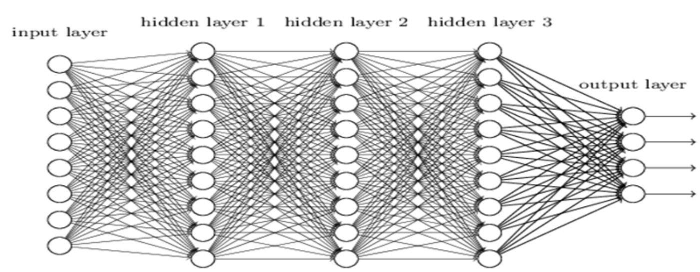
Arjun

Deep Learning

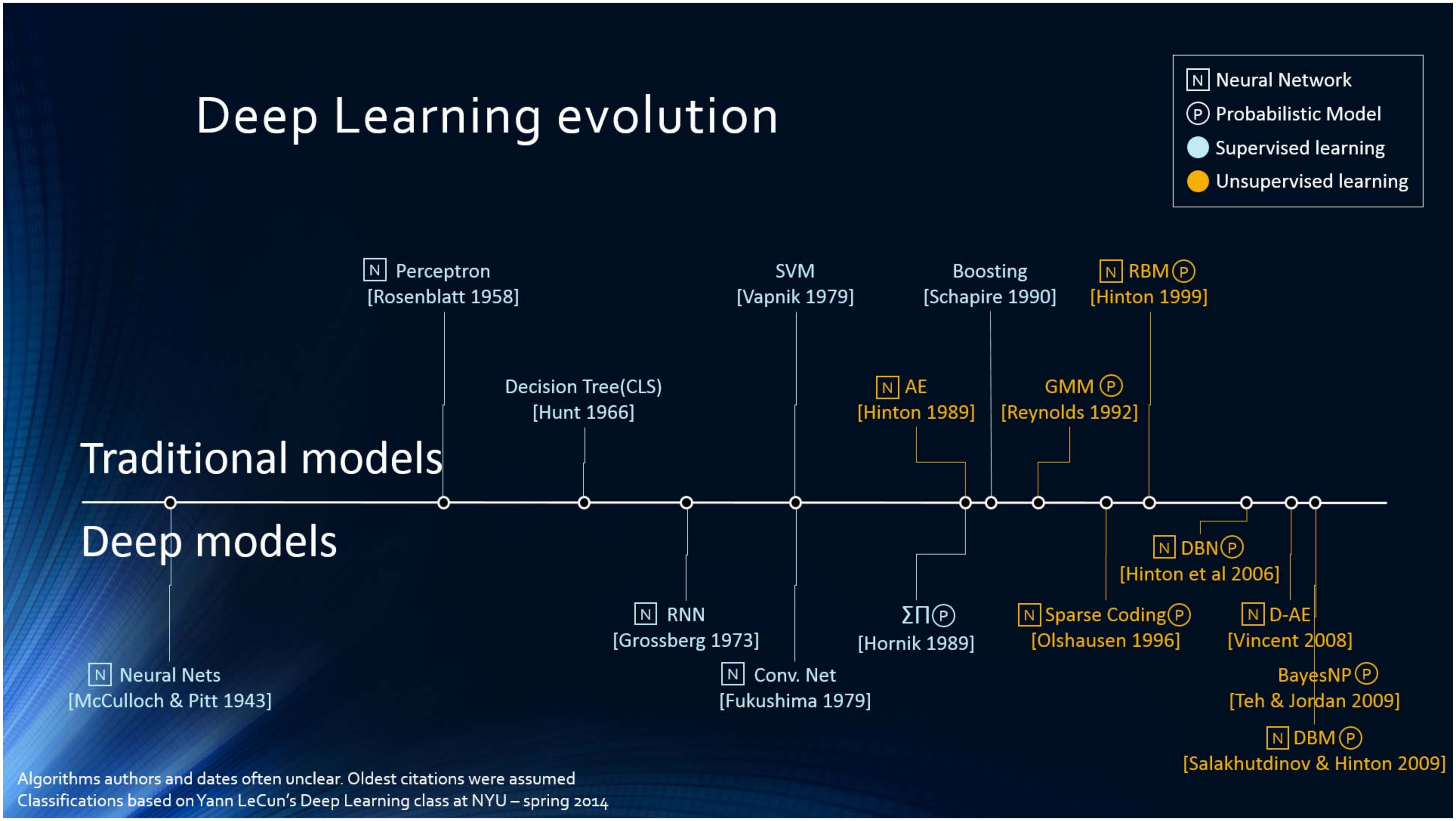
How do you guide the model to find the best features?

$N \times N$

NEURAL NETWORK



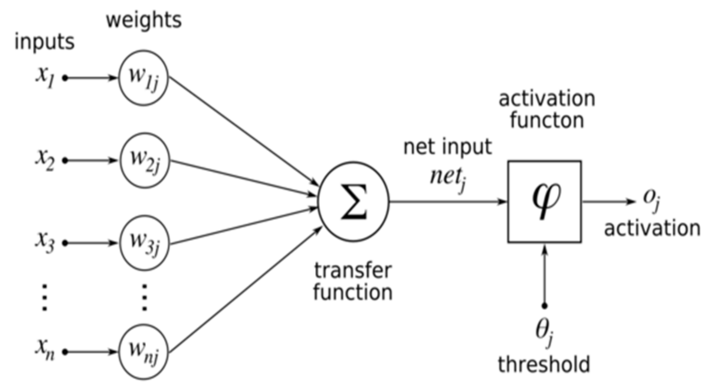
Arjun


Deep Learning evolution
N Neural Network
P Probabilistic Model
Supervised learning
Unsupervised learning
Traditional models
Deep models
N Perceptron [Rosenblatt 1958]
Decision Tree(CLS) [Hunt 1966]
SVM [Vapnik 1979]
N AE [Hinton 1989]
Boosting [Schapire 1990]
GMM P [Reynolds 1992]
N RBM P [Hinton 1999]
N Neural Nets [McCulloch & Pitt 1943]
N RNN [Grossberg 1973]
N Conv. Net [Fukushima 1979]
ΣΠ P [Hornik 1989]
N Sparse Coding P [Olshausen 1996]
N DBN P [Hinton et al 2006]
N D-AE [Vincent 2008]
BayesNP P [Teh & Jordan 2009]
N DBM P [Salakhutdinov & Hinton 2009]
Algorithms authors and dates often unclear. Oldest citations were assumed
Classifications based on Yann LeCun's Deep Learning class at NYU – spring 2014

Biologically Inspired Artificial Neurons

Neurons and biological brains

- Each Human has a 100 Billion neurons. Each Neuron connects to 10k other neurons ~ 1 Trillion Synapses
- Neurons send and receive signals. A neuron "fires" if the summation of its input is above a threshold



Simple Artificial Neuron or "Switch" or (Single Layer) Perceptron

- Has a number of "weighted" inputs
- Considers sum of weighted inputs and "fires" an output based on it's activation function

Multi Layer Perceptrons or Artificial Neural Networks (ANNs)

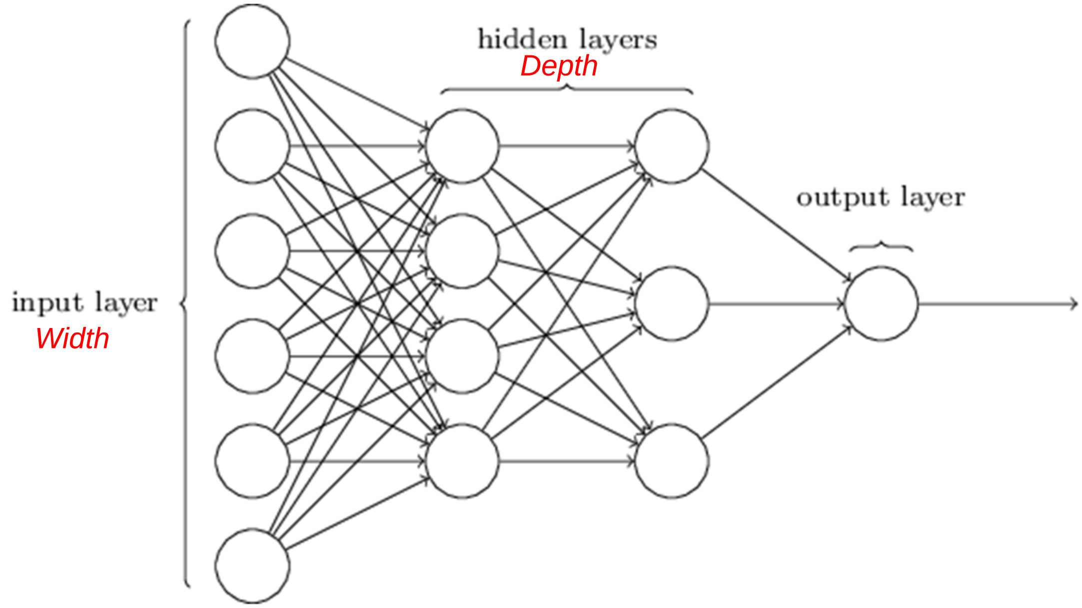
Group together multiple Artificial Neurons into a Network

- Single Layer Perceptrons can be used to represent linearly separable functions

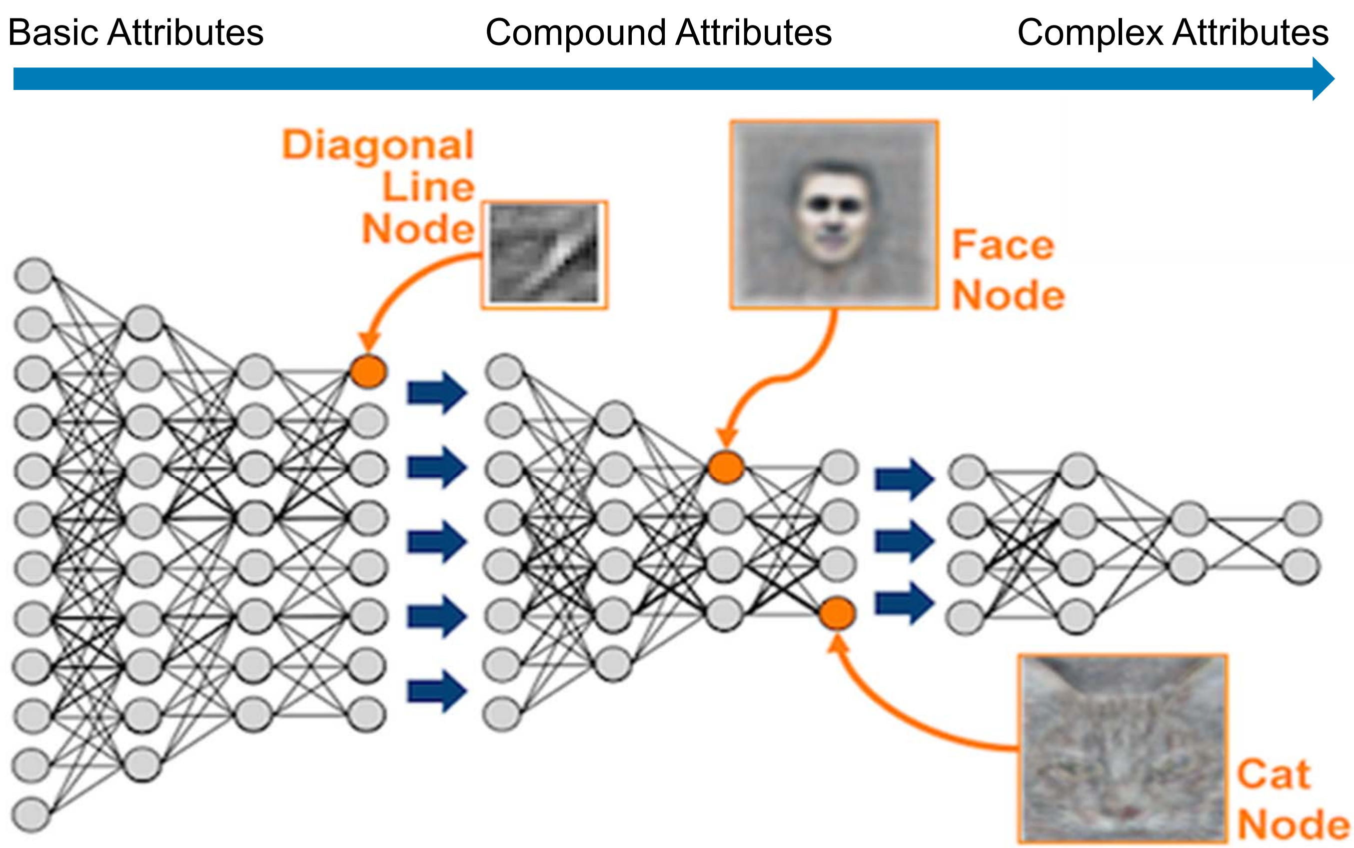
- However many problems are not linearly separable, may be represented as 2D or 3D spaces (*Lippmann in the 1987 paper "An introduction to computing with neural nets")*

- *Generally arrive at these parameters through experimentation*

- *Architecture is based on the layout of various layers E.g. AlexNet, ImageNet*

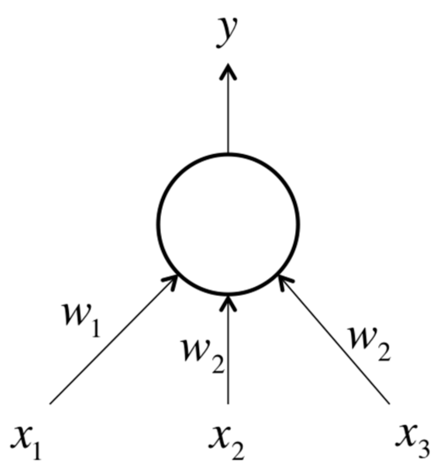


Deep Neural Network – How do they work?



DELL EMC

A Simple Example: Supervised Learning



$$y^{(i)} = w_1 x_1^{(i)} + w_2 x_2^{(i)} + w_3 x_3^{(i)}$$

$$z = \sum_k w_k x_k$$

$$y = \frac{1}{1 + e^{-z}}$$

$z^{(1)} = w_1 x_1^{(1)} + w_2 x_2^{(1)} + w_3 x_3^{(1)}$
$z^{(2)} = w_1 x_1^{(2)} + w_2 x_2^{(2)} + w_3 x_3^{(2)}$
$z^{(3)} = w_1 x_1^{(3)} + w_2 x_2^{(3)} + w_3 x_3^{(3)}$

Error criteria: Minimize the square error over all of the training examples

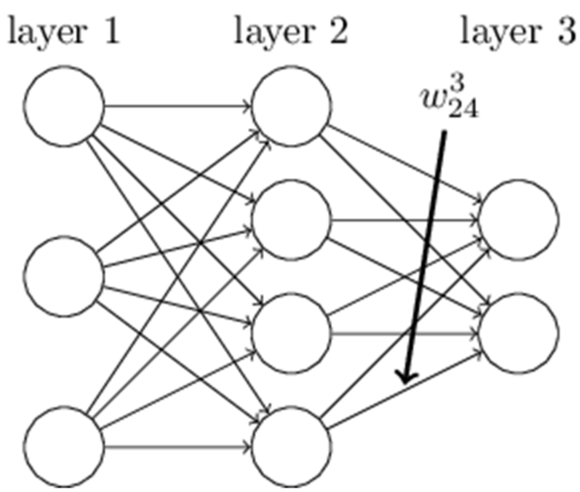
Minimise $E = \frac{1}{2} \sum_i (t^{(i)} - y^{(i)})^2$

Where $t^{(i)}$ is the true answer for the i^{th} training example

To modify the weights:

$$\Delta w_k = \sum_i \epsilon x_k^{(i)} y^{(i)} \left(1 - y^{(i)}\right) \left(t^{(i)} - y^{(i)}\right)$$

Training and Backpropagation (Supervised learning)



The backpropagation algorithm was originally introduced in the 1970s, but its importance wasn't fully appreciated until a famous 1986 paper by David Rumelhart, Geoffrey Hinton, and Ronald Williams

Fast Matrix Maths to the rescue

$$(\mathrm{W}^{l+1})^T \delta^{l+1} = \begin{bmatrix} w_{11}^{l+1} & \cdots & w_{m1}^{l+1} & \cdots & w_{M1}^{l+1} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{1k}^{l+1} & \cdots & w_{mk}^{l+1} & \cdots & w_{Mk}^{l+1} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{1K}^{l+1} & \cdots & w_{mK}^{l+1} & \cdots & w_{MK}^{l+1} \end{bmatrix} \begin{bmatrix} \delta_1^{l+1} \\ \vdots \\ \delta_m^{l+1} \\ \vdots \\ \delta_M^{l+1} \end{bmatrix} \begin{bmatrix} \sum_{m=1}^{M} \delta_m^{l+1} w_{m1}^{l+1} \\ \vdots \\ \sum_{m=1}^{M} \delta_m^{l+1} w_{mk}^{l+1} \\ \vdots \\ \sum_{m=1}^{M} \delta_m^{l+1} w_{mK}^{l+1} \end{bmatrix} \sim \delta^l$$

1. Do forward propagation (matrix operation)
2. Calculate Error of the final layer
3. Do backward propagation (matrix operation)
4. Update the weights

Deep Learning in practice

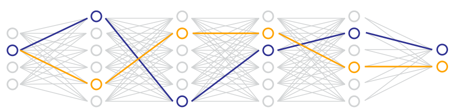
- **Step 1: Training**

(In Data Center – Over Hours/Days/Weeks)

Lots of labeled input data

Person

Create "Deep neural net" math model



Output Classification

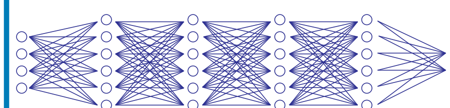
90% person
8% traffic light

Trained Model

- **Step 2: Scoring**

(End point or Data Center - Instantaneous)

New input from camera and sensors



Trained neural network model

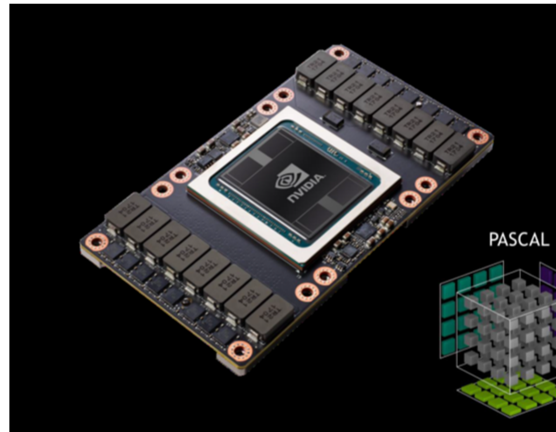
97% person

Output Classification

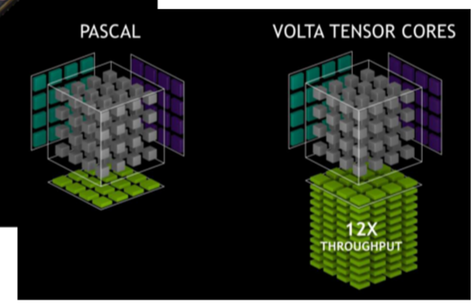
Internal Use - Confidential

D&LLEMC

Scaling DL: Compute

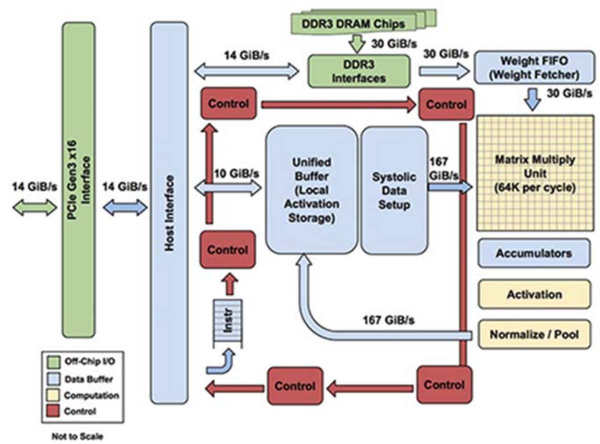


NVIDIA TESLA V100
- 640 NEW Tensor cores
- 300 GB/s NVLink2



Compute Efficiency
- Matrix operations are important (GEneral Matrix to Matrix)
    - GEMM Processor friendly but need a lot of memory
- Floating point precision not critical – Int8, Int16 are sufficient
- Data transport:
    - High Bandwidth DRAM and memory hierarchy
- Dataflow architectures, memristors, Op AMPS

Scale Up vs Scale Out
- On-node vs off-node
- Faster off-chip interconnects
- Parallel Programming frameworks e.g. MPI
- BitFusion.io for GPU/FPGA Virtualization
- Distributed Deep Learning by IBM – GPUs over IB



Google TPU

Matrix Multiplier Unit (MXU): 65,536 8-bit multiply-and-add units for matrix operations

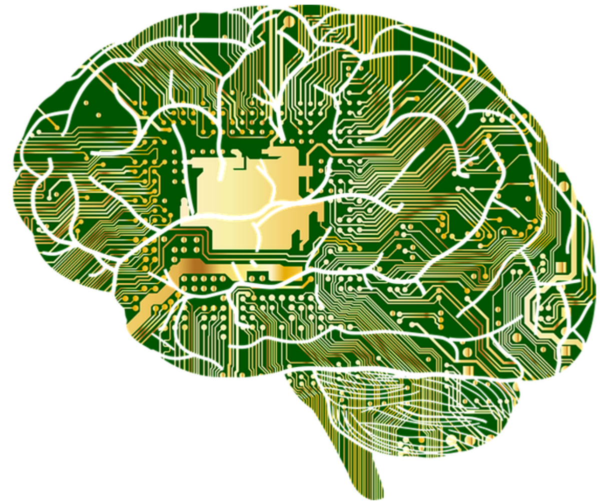
Internal Use - Confidential

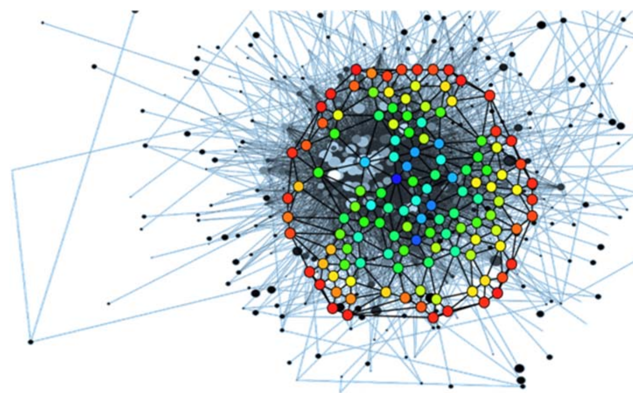
D&LLEMC

Accelerating DL: Productivity

Software infrastructure:

- Partitioning problem space efficiently across nodes
- More efficient algorithms for doing the forward and back propagation (optimized algorithms and matrix computations)
- Distributed compute infrastructure that allows you to efficiently scale-out
- A fast (distributed) filesystem for managing data
- Resilient programming model, easy to deploy and maintain





Summary

- Many of the techniques developed for HPC are already making their way into the latest DL processors
- Newer techniques unique to AI are also being developed, like optimized matrix calculators using memristors
- Scale-out and modularity will dominate provided we create the proper SW frameworks – higher speed interconnects will remain crucial
- Practicality of the technology will ultimately determine which technologies become mainstream

References

https://petewarden.com/2015/04/20/why-gemm-is-at-the-heart-of-deep-learning/

M. Abadi et al. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. (2015). http://www.tensorflow.org

A. F. Aji and K. Heafield. 2017: Sparse Communication for Distributed Gradient Descent. (2017). arXiv:1704.05021

F. Akopyan et al. 2015. TrueNorth: Design and Tool Flow of a 65 mW 1 Million Neuron Programmable Neurosynaptic Chip. IEEE Transactions on ComputerAided Design of Integrated Circuits and Systems 34, 10 (2015), 1537–1557.

TAL BEN-NUN and TORSTEN HOEFLER, ETH Zurich, 2018: Demystifying Parallel and Distributed Deep Learning: An In-Depth Concurrency Analysis (2018) arXiv:1802.09941v2

Questions or Feedback: Adnan.Khaleel@Dell.com