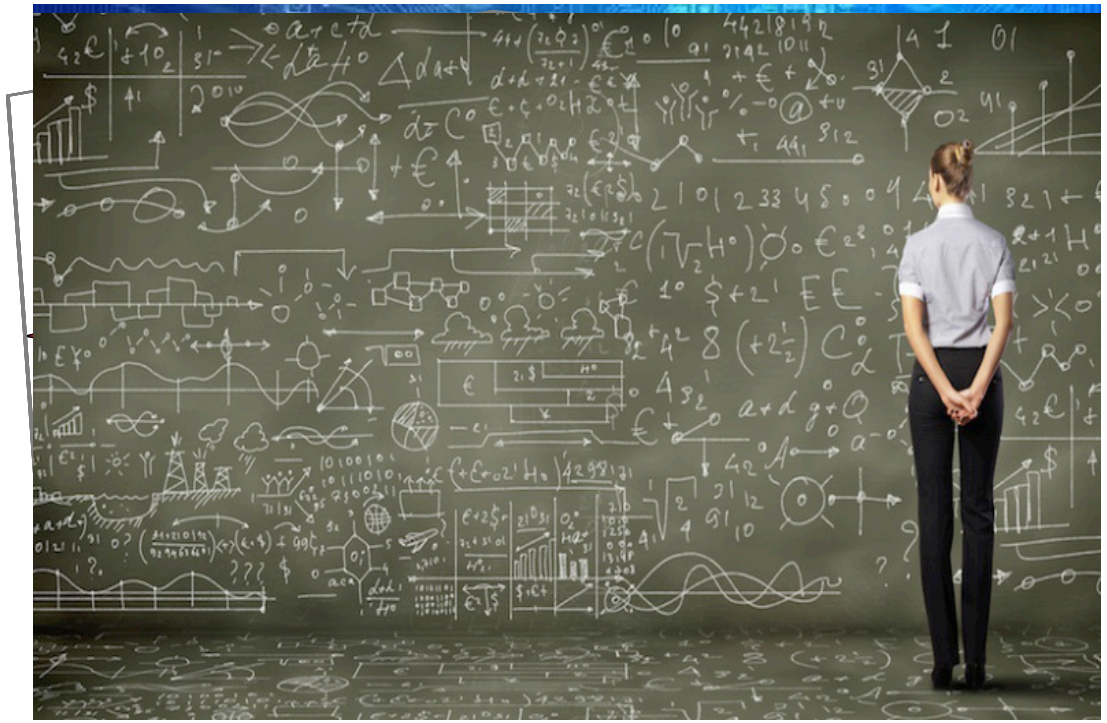
Applying Recent Advances in Machine Learning to Networking



David Meyer
Brocade Chief Scientist and Fellow
dmm@{brocade.com,uoregon.edu,1-4-5.net,..}
http://www.1-4-5.net/~dmm/ml/talks/2016/ngp_forum.{pptx,pdf}
NGP Forum/SDN & Openflow World Congress
10-14 October 2016
The Hague, Netherlands

You might be surprised but what is going to drive innovation in the enterprise and in the public cloud is machine learning.

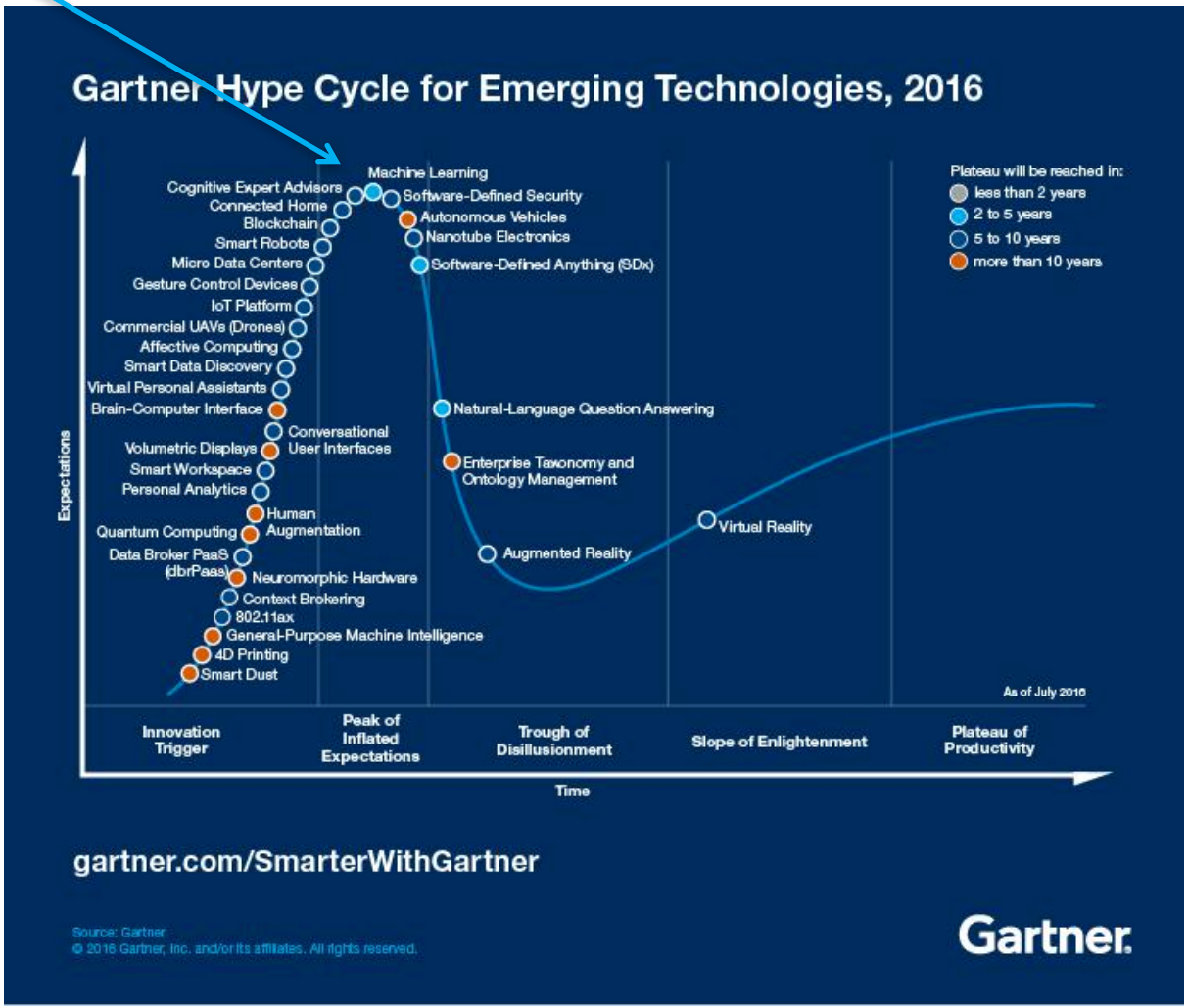
Bill Coughran, Sequoia Capital #ONUGSpring16

Machine Learning is the way we are going to automate your automation.

Chris Wright, RedHat CTO #RHSummit

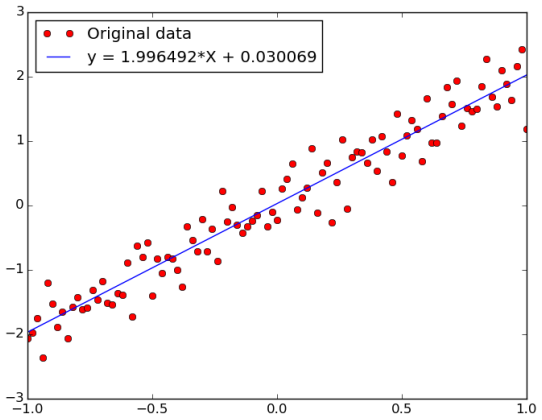
Whatever Machine Learning Is…

You are here (again)



Agenda

- What Is Machine Learning?
- Why All The (Machine Learning) Excitement?
- Combining SDN and Machine Learning
- What's On the Horizon?
- Technical explanations/code
  - https://github.com/davidmeyer/ml
  - http://www.1-4-5.net/~dmm/ml



What Is Machine Learning?

The complexity in traditional computer programming is in the code (programs that people write). In machine learning, learning algorithms are in principle simple and the complexity (structure) is in the data. Is there a way that we can automatically learn that structure? That is what is at the heart of machine learning.

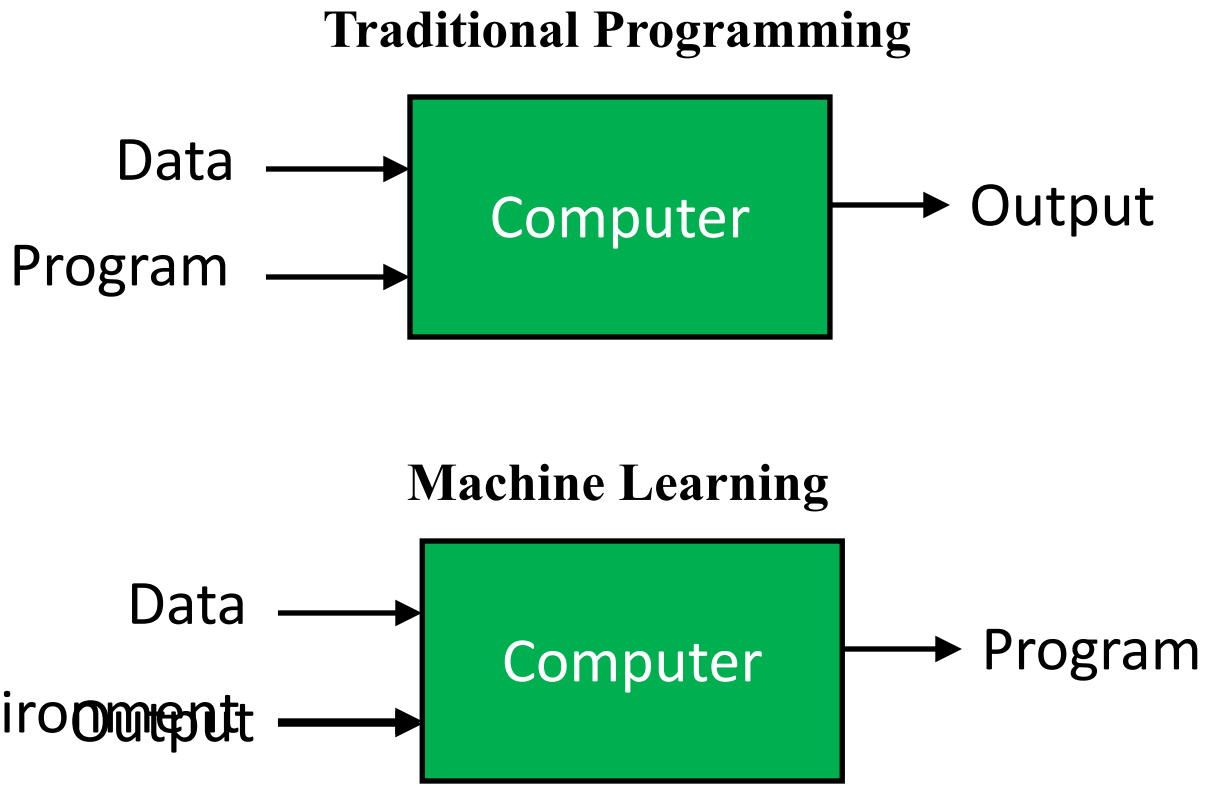
-- Andrew Ng

- Said another way, we want to discover the *Data Generating Distribution* that underlies the data that we observe. This is the function that we want to learn.

- Moreover, we care about primarily about the generalization accuracy of our model (function)
  - *Accuracy on examples we have not yet seen (BTW, how is this possible?)*
  - as opposed the accuracy on the training set (note: overfitting)

Same Thing Said In Cartoon Form

(Three main types of Machine Learning)



So When Would We Use Machine Learning?

- When patterns exists in our data
  - Even if we don't know what they are
    - Or perhaps especially when we don't know what they are
    - Or if they are just noise
- We can not pin down the functional relationships mathematically
  - Else we would just code up the algorithm
  - Neural networks as function approximators
    - Need this for scale
- When we have lots of (unlabeled) data
  - Labeled training sets harder to come by
  - Data is of high-dimension
    - High dimension "features"
    - For example, sensor data
  - Want to "discover" lower-dimension representations
    - Dimension reduction
- Aside: Machine Learning is heavily focused on implementability
  - Frequently using well know numerical optimization techniques
  - Lots of open source code available
    - All kinds of open source frame works, e.g., https://www.tensorflow.org/ or http://torch.ch/ or …
    - Well-worn libraries http://scikit-learn.org/stable/ (many others)

Examples of Machine Learning Problems

- Pattern Recognition
  - Facial identities or facial expressions
  - Handwritten or spoken words (e.g., Siri)
  - Medical images
  - Sensor Data/IoT
  - Recommender Systems
- Optimization
  - Many parameters have “hidden” relationships that can be the basis of optimization
- Pattern Generation
  - Generating images or motion sequences
- Anomaly Detection
  - Unusual patterns in the telemetry from physical and/or virtual plants (e.g., data centers)
  - Unusual sequences of credit card transactions
  - Unusual patterns of sensor data from a nuclear power plant
    - or unusual sound in your car engine or …
- Prediction
  - Future stock prices or currency exchange rates
  - Network events
  - …

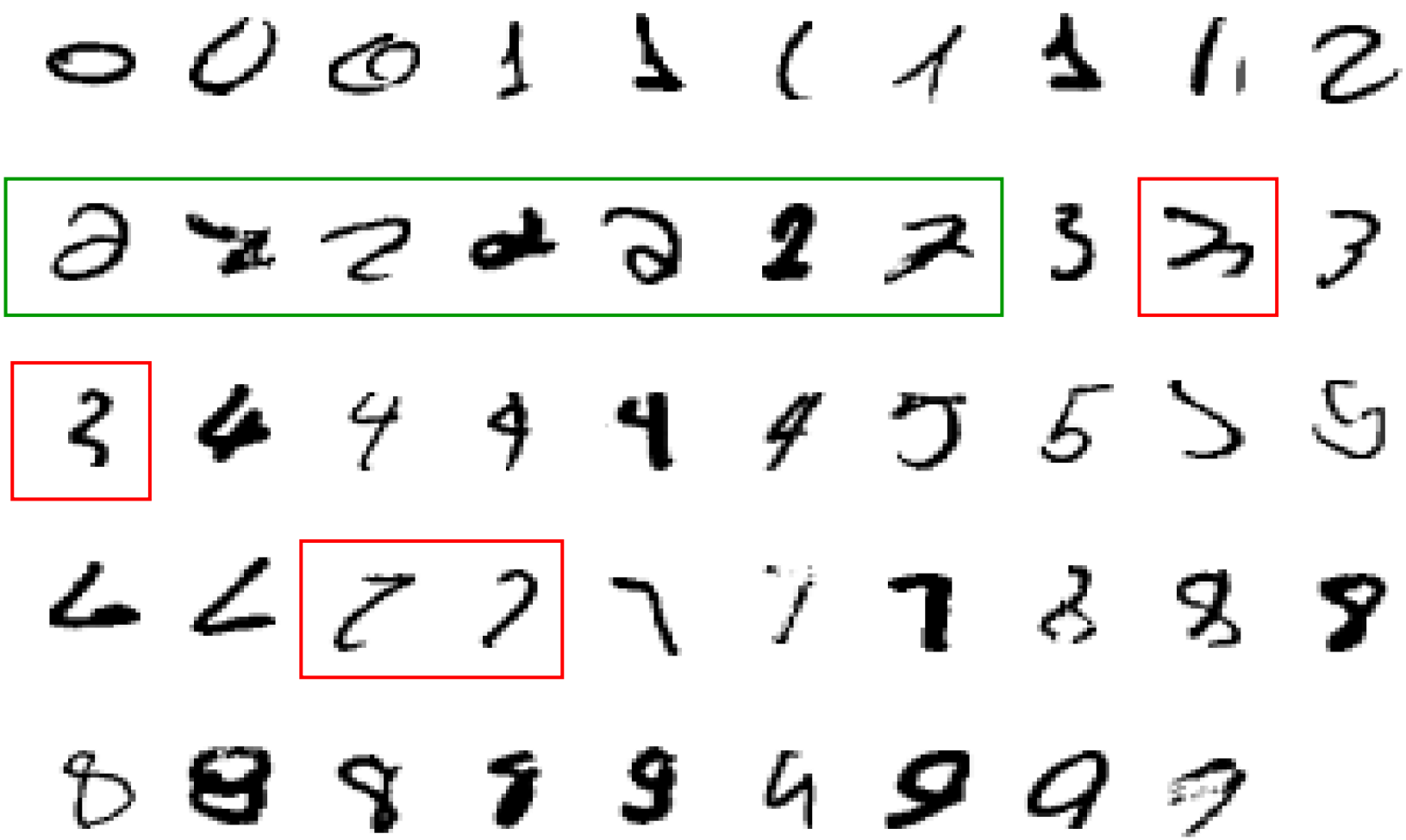
BTW, Why Is Machine Learning Hard?

You See

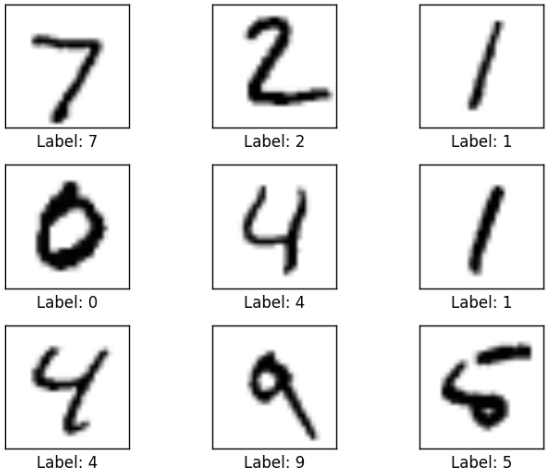
Your ML Algorithm Sees
A Bunch of Bits

Why Machine Learning Is Hard, Redux

What is a “2”?



BTW, How Hard Is It *These Days* To Write Code That Recognizes Handwritten Digits?



MNIST: 28x28 Grayscale Handwritten Digit Dataset

- Training set: 55000 images
- Test set: 10000 images
- Validation set: 5000 images

Features here are pixels (28x28 = 784)

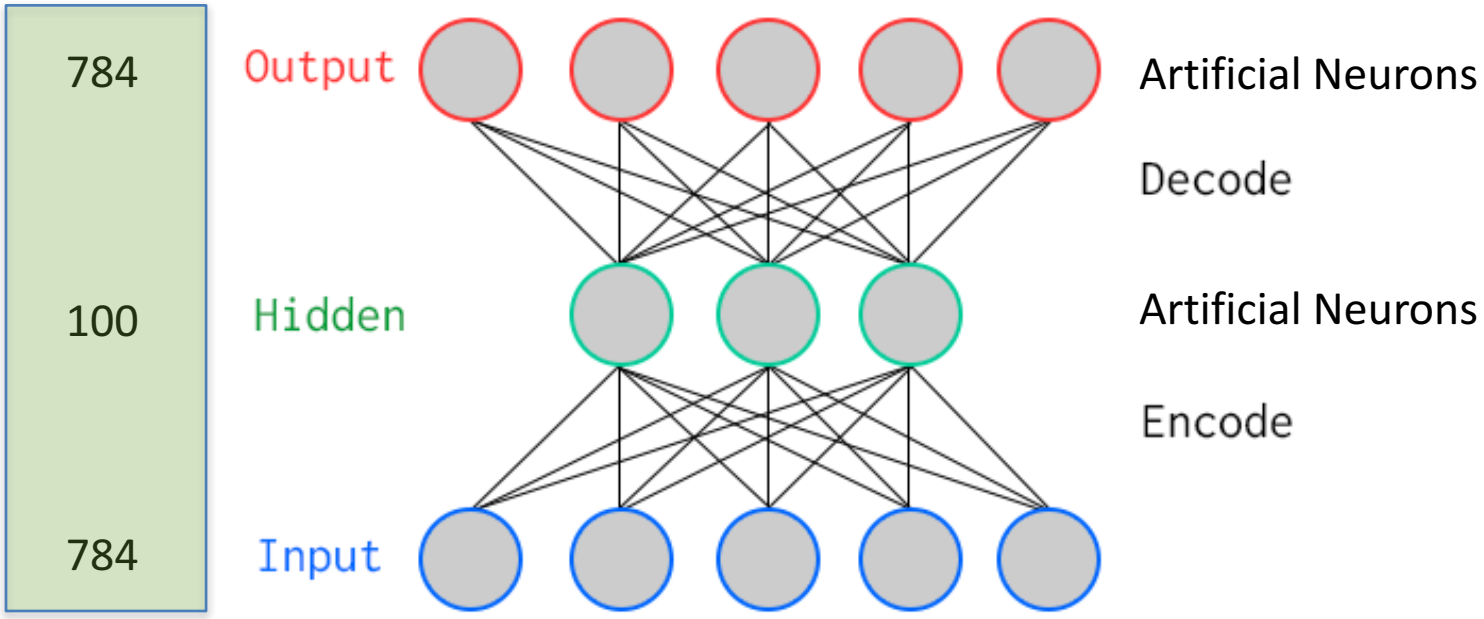
Can We Analyze Network Flow Data And Other KPIs In A Similar Fashion?

Src IP	Dst IP	Appln	Src Port	Dst Port	Protocol	DSCP	TCP FLAGS	Flow Rate	Traffic	Packets	NextHop	FNF NbarApp
192.168.10.	192.168.13.1	compressnet	2	169	TCP	AF12	UAP SF	0 Kbps	1.0 KB	2	15.80.39.28	-
192.168.10.	192.168.13.1	compressnet	2	564	TCP	AF12	UAP SF	0 Kbps	1.0 KB	2	190.23.69.213	-
192.168.10.	192.168.13.1	compressnet	2	741	TCP	AF12	UAP SF	0 Kbps	1.0 KB	2	95.55.61.159	-
192.168.10.	192.168.13.1	compressnet	281	2	TCP	AF12	UAP SF	1.0 Kbps	1.0 KB	2	223.191.78.79	-
192.168.10.	192.168.13.1	compressnet	165	3	TCP	AF12	UAP SF	0 Kbps	1.0 KB	2	64.15.70.93	-
192.168.10.	192.168.13.1	compressnet	424	3	TCP	AF12	UAP SF	0 Kbps	1.0 KB	2	149.191.44.148	-
192.168.10.	192.168.13.1	compressnet	822	3	TCP	AF12	UAP SF	0 Kbps	1.0 KB	2	73.7.175.22	-
192.168.10.	192.168.13.1	rje	800	5	TCP	AF12	UAP SF	0 Kbps	1.0 KB	2	74.157.168.220	-
192.168.10.	192.168.13.1	discard	9	714	TCP	AF12	UAP SF	0 Kbps	1.0 KB	2	1.209.56.6	-
192.168.10.	192.168.13.1	daytime	13	252	TCP	AF12	UAP SF	0 Kbps	1.0 KB	2	232.237.195.100	-
192.168.10.	192.168.13.1	daytime	960	13	TCP	AF12	UAP SF	0 Kbps	1.0 KB	2	22.88.95.248	-
192.168.10.	192.168.13.1	msp	18	116	TCP	AF12	UAP SF	1.0 Kbps	1.0 KB	2	81.203.252.131	-
192.168.10.	192.168.13.1	msp	18	735	TCP	AF12	UAP SF	0 Kbps	1.0 KB	2	18.50.17.126	-
192.168.10.	192.168.13.1	ftp-data	20	513	TCP	AF12	UAP SF	0 Kbps	1.0 KB	2	240.7.195.4	-

Features here are various counters and other KPIs

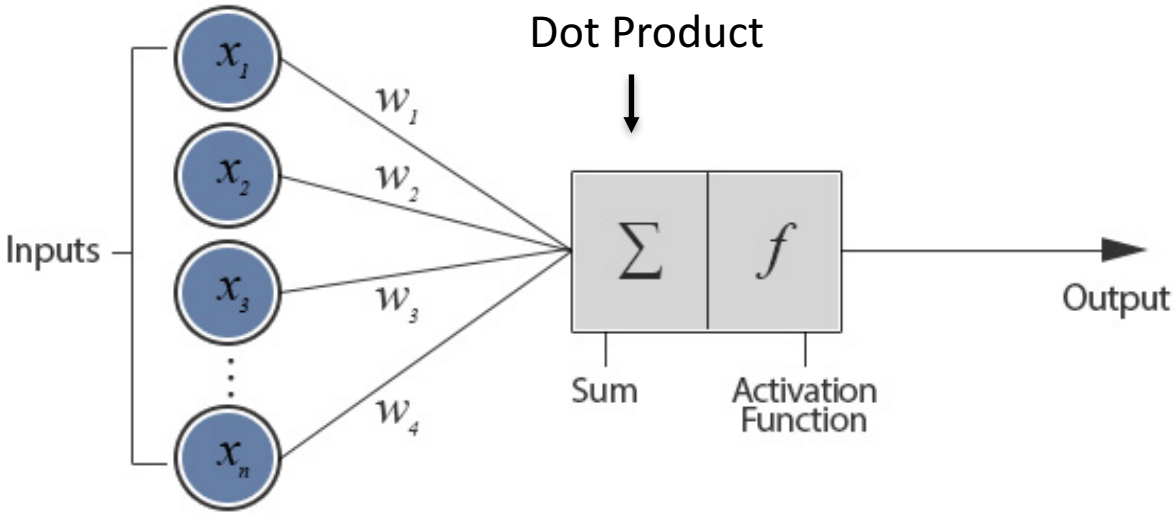
One Way To Learn To Recognize MNIST: Use An Autoencoder

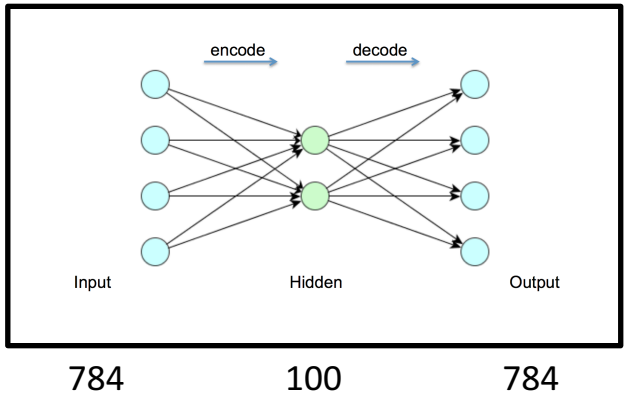
Special Kind of Neural Network



- Key Characteristic: Hidden layer has fewer units than input/output → Compression
- Goal: Minimize reconstruction (decode) error
  - How to define error (loss, cost)?
  - Binary classification: Threshold reconstruction error → normal/abnormal
- Unsupervised learning

But First: What Does A Single Neuron Do?



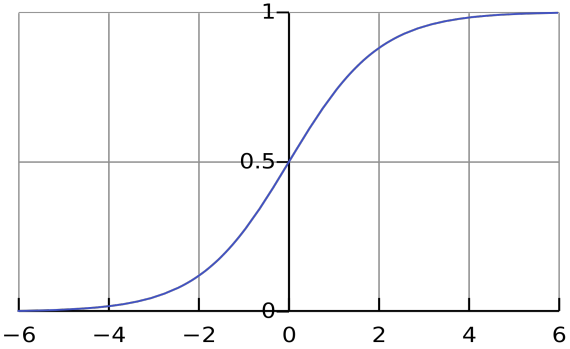


BTW, how many parameters?

784x100 + 100x784 = 156800

$$\boldsymbol{W}^T\boldsymbol{x} = \begin{bmatrix} w_1 & w_2 & \cdots & w_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \sum_{i=1}^{n} w_i x_i = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n$$

$$f(\boldsymbol{W}^T\boldsymbol{x} + \boldsymbol{b}) = \sigma(\boldsymbol{W}^T\boldsymbol{x} + \boldsymbol{b}) = \frac{1}{1 + e^{-(\boldsymbol{W}^T\boldsymbol{x} + \boldsymbol{b})}}$$



All Cool, But How Does The Autoencoder Actually Work?

- Encoder $h^{(t)} = f_\theta(x^{(t)}), \{x^{(1)}, \ldots, x^{(T)}\}$

Where h is feature vector or representation or code computed from x

- Decoder

maps from feature space back into input space, producing a reconstruction

$$r = g_\theta(h)$$

attempting to incur the lowest possible reconstruction error $L(x, r)$.

Good generalization means low reconstruction error at test examples, while having high reconstruction error for most other x configurations

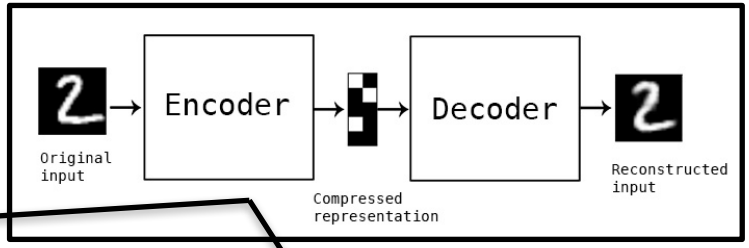
$$h^{(i)} = f_{\theta_e}(x^{(i)}) = \sigma(\theta_e^{\mathrm{T}} x^{(i)} + b_e^{(i)})$$
$$r^{(i)} = g_{\theta_r}(h^{(i)}) = \sigma(\theta_r^{\mathrm{T}} h^{(i)} + b_r^{(i)})$$

$$L(x, r) = \frac{1}{T} \sum_{i=1}^{T} (x^{(i)} - r^{(i)})^2$$

How Hard To Code This Up In Tensorflow?

```
#
#       encoder/decoder
#
#       try tf.nn.sigmoid, tf.nn.relu, etc for nonlinearity
#       nonlinearity=False means transfer function (aka activation funtion)
#       g(x) = x
#
def encoder(x, nonlinearity=False):
    code = tf.add(tf.matmul(x, weights['encoder']), biases['encoder'])
    if nonlinearity:
        code = nonlinearity(code)
    return code

def decoder(code, nonlinearity=False):
    reconstruction = tf.add(tf.matmul(code, weights['decoder']), biases['decoder'])
    if nonlinearity:
        reconstruction = nonlinearity(reconstruction)
    return reconstruction
#
#       get the encoding and decoding operations
#
#       relu seems less efficient here
#
#
#       first encode
#
encoder_op = encoder(X,tf.nn.sigmoid)
#
#       then decode
#
decoder_op = decoder(encoder_op,tf.nn.sigmoid)
#
#       decoder_op is our predicted value (y_pred)
#
y_pred = decoder_op
#
#       y_true is the input X
#
y_true = X
#
reg_losses   = tf.get_collection(tf.GraphKeys.REGULARIZATION_LOSSES)
reg_constant = 0.01
#
if (USE_REGULARIZER):
        error = tf.add(tf.reduce_mean(tf.square(tf.sub(y_true,y_pred))),
                       tf.mul(reg_constant,tf.reduce_sum(reg_losses)))
else:
        error = tf.reduce_mean(tf.square(tf.sub(y_true,y_pred)))
```



$$h^{(i)} = f_{\theta_e}(x^{(i)}) = \sigma(\theta_e^{\mathrm{T}} x^{(i)} + b_e^{(i)})$$

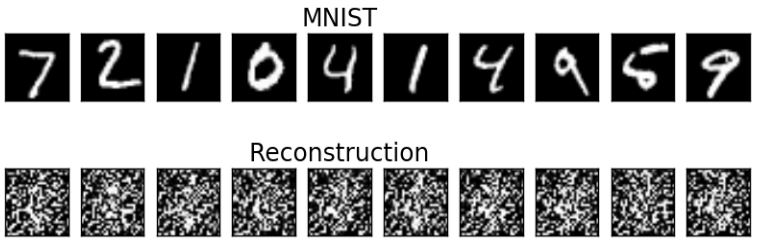
$$r^{(i)} = g_{\theta_r}(h^{(i)}) = \sigma(\theta_r^{\mathrm{T}} h^{(i)} + b_r^{(i)})$$

$$L(x, r) = \frac{1}{T} \sum_{i=1}^{T} (x^{(i)} - r^{(i)})^2$$

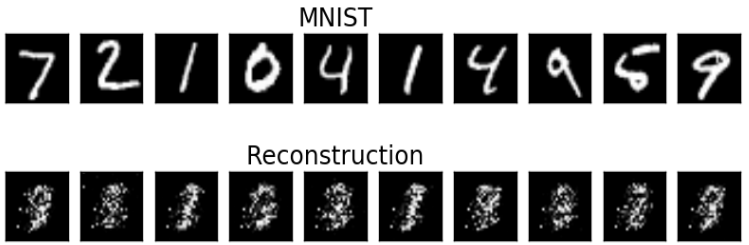
https://github.com/davidmeyer/ml/blob/master/tensorflow/autoencoder.{py,ipynb}

Autoencoder Output

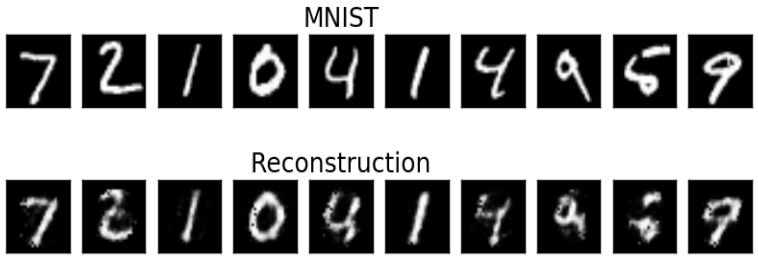
1 example



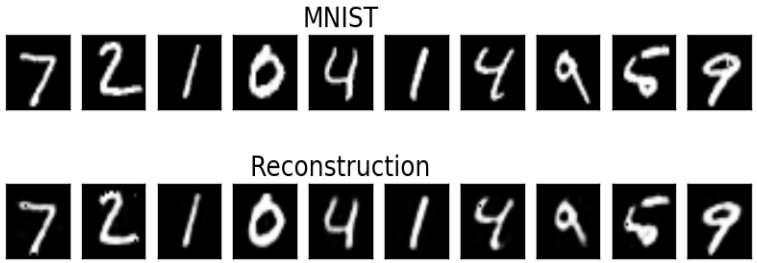
10 examples



100 examples



1000 examples



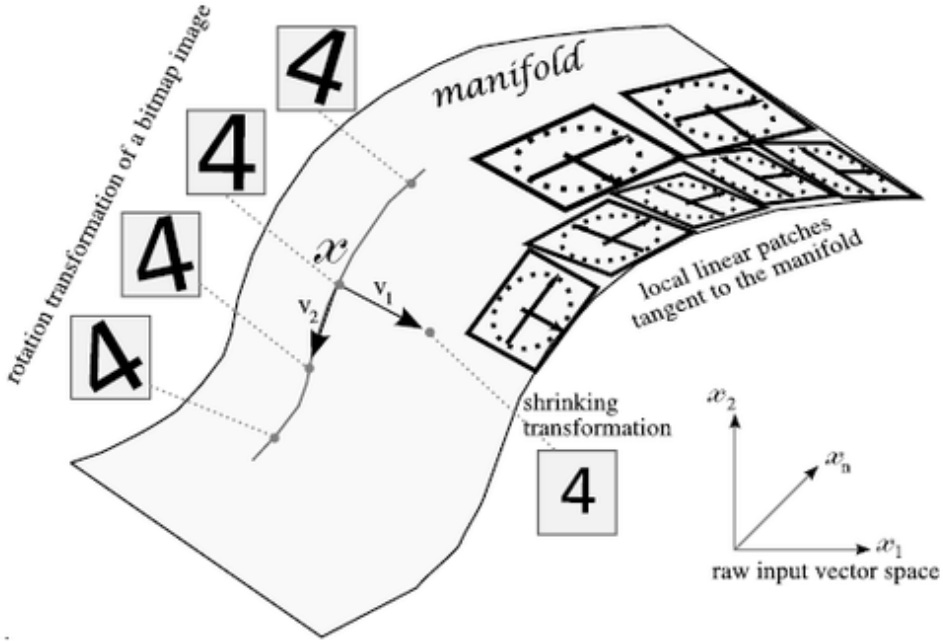
After training, the AE gets low *reconstruction error* on digits from MNIST and high reconstruction error on everything else: It has learned to recognize MNIST

$$L(x, r) = \frac{1}{T} \sum_{i=1}^{T} (x^{(i)} - r^{(i)})^2$$

What Is Really Going On Here?

(Manifold Hypothesis)

- X is a high dimensional vector ($28^2 = 784$)
- Data concentrated around a lower dimensional manifold



- Hope to find representation *R* of that manifold

Graphic courtesy http://www.deeplearningbook.org

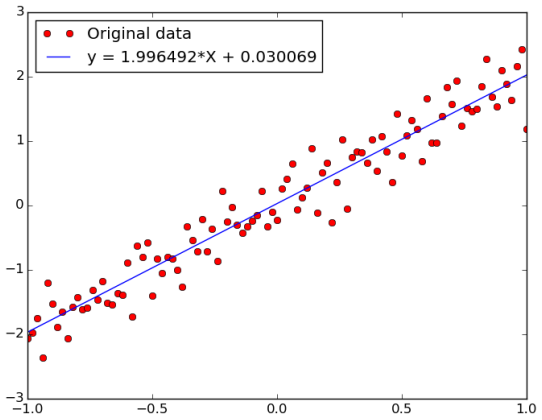
Many Beautiful Mysteries Remain

- Back-propagation
  - Gradient-based optimization with gradients computed by back-prop
  - Why is such a simple algorithm so powerful?
  - Optimization an active area of research
  - Many new techniques , e.g., Layer Normalization
  - https://arxiv.org/pdf/1607.06450v1.pdf
- Neural Nets
  - What are the units (artificial neurons) actually doing?
  - https://arxiv.org/pdf/1509.06321.pdf
- Adversarial Images
  - Generative Adversarial Nets (GANs)
  - https://arxiv.org/abs/1412.6572
- Why does deep and cheap learning work so well?
  - Physics perspective
  - http://arxiv.org/pdf/1608.08225v1.pdf
- …

- BTW, how many 28x28 grayscale images are there?
- Say there are 8 bits of grayscale → 256^{784} possible images
- How can such a simple autoencoder recognize them?

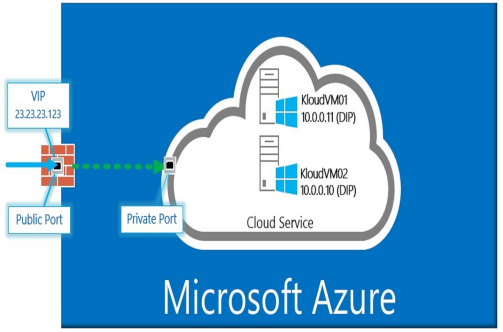
Agenda

- What Is Machine Learning?
- Why All The (Machine Learning) Excitement?
- Combining SDN and Machine Learning
- What's On the Horizon?
- Technical explanations/code
  - https://github.com/davidmeyer/ml
  - http://www.1-4-5.net/~dmm/ml



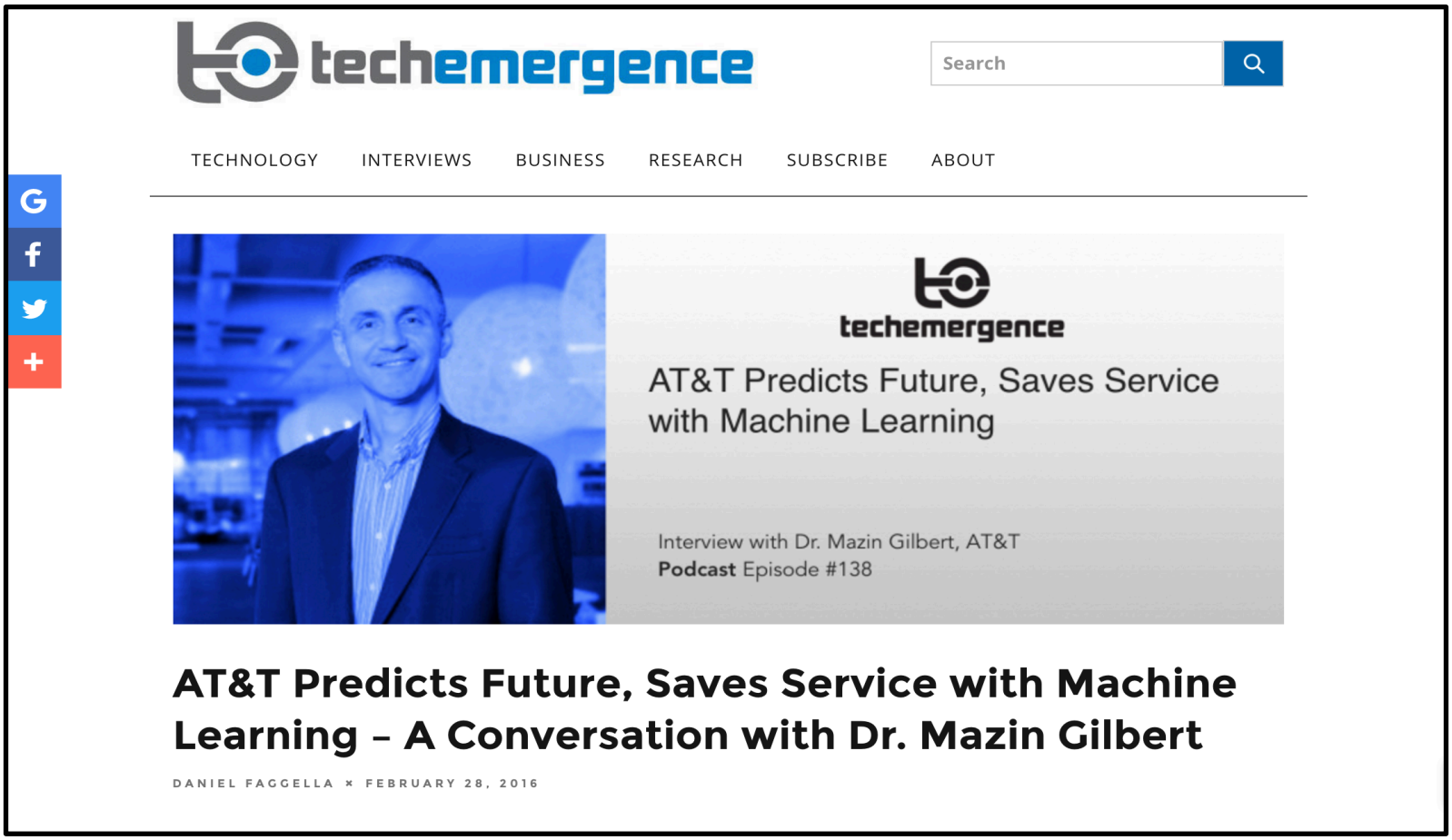
Why All The ML Excitement?

ML Applications You Likely Interact With Everyday



Why this is relevant: Compute, Storage, Networking, Security and Energy (CSNSE) use cases will all use similar technology (deep nets, …)

While ML For Networking Is Still In Its Infancy…

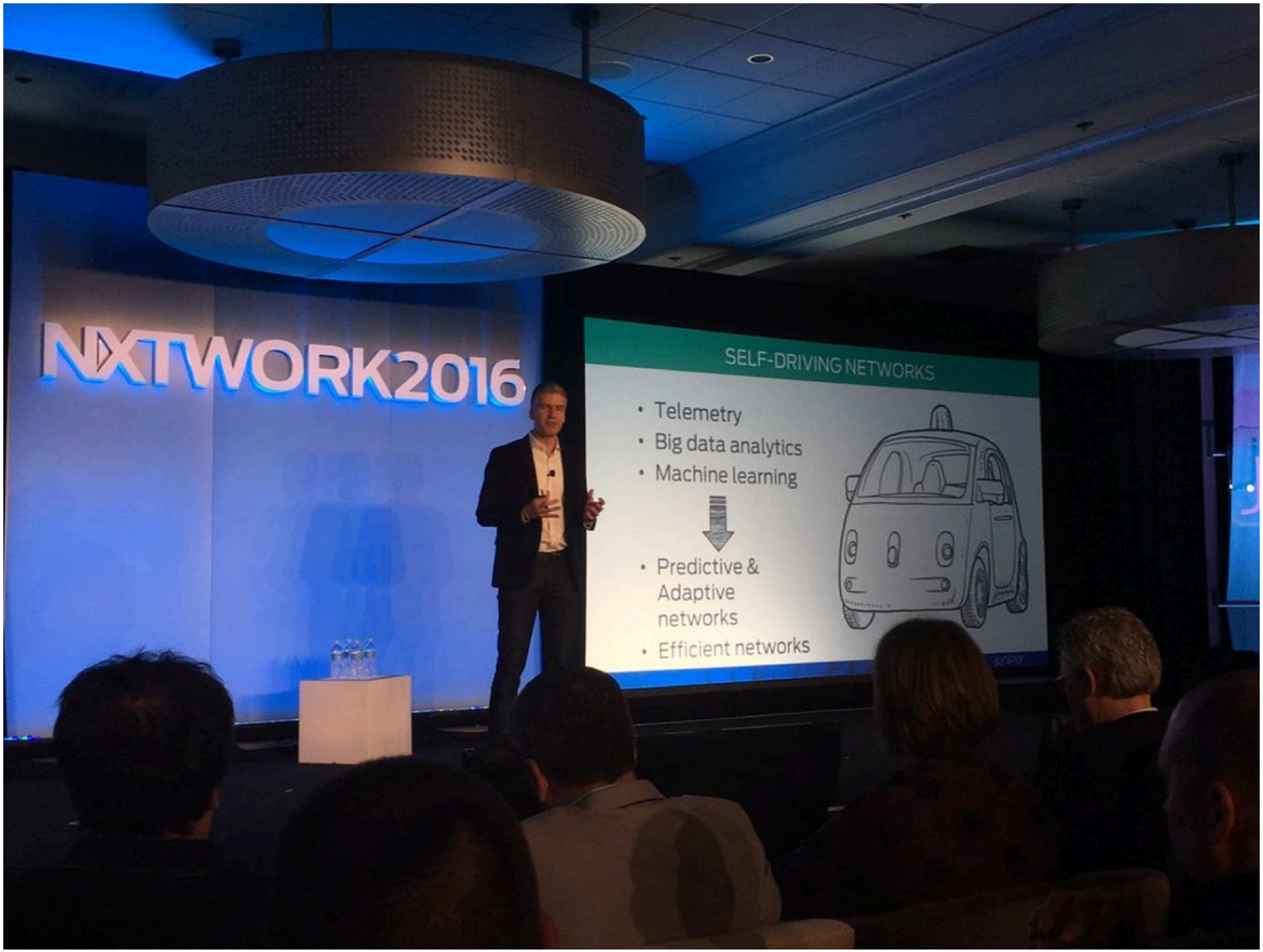


Machine learning is revolutionizing the world as we know it, both in and out of the digital realms, and is predicted to expand to a $2 trillion market by 2025.

5G Networks


CogNet
Home
Project
CogNet in 5GPPP
Dissemination
Events
News
Contact
Building an Intelligent System
of Insights and Action
for 5G Network Management
COGNET

Vendors Joining The Game As Well



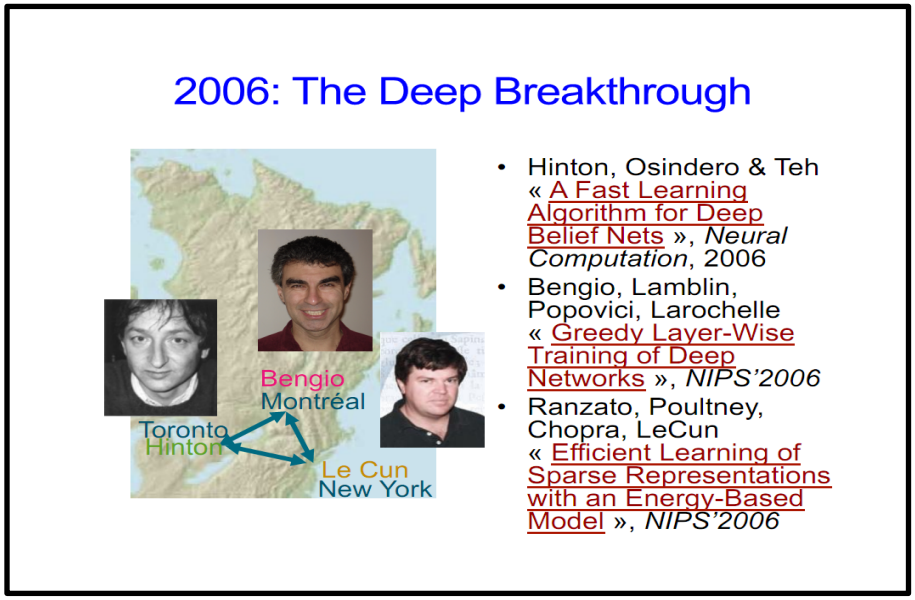
Why is this all happening now?

- **Before 2006 people thought deep neural networks couldn't be trained**
  - So why now?

- **Theoretical breakthroughs in 2006**
  - Learned how to train deep neural networks
    - Technically: RBMs, Stacked autoencoders, sparse coding
  - Nice overview of LBH DL journey: http://chronicle.com/article/The-Believers/190147/

- **Compute**
  - CPUs were 2^20s of times too slow
  - Parallel processing/algorithms
  - GPUs + OpenCL/CUDA
  - FPGAs, custom hardware

- **Datasets**
  - Massive data sets: Google, FB, Baidu, …
  - *Standardized* data sets



- **Alternate view of history?**
  - LBH Nature DL review: http://www.nature.com/nature/journal/v521/n7553/full/nature14539.html
  - Jürgen Schmidhuber's critique : http://people.idsia.ch/~juergen/deep-learning-conspiracy.html
  - LBH rebuttal: http://recode.net/2015/07/15/ai-conspiracy-the-scientists-behind-deep-learning/

Image courtesy Yoshua Bengio

BTW, What Kinds of Custom H/W?

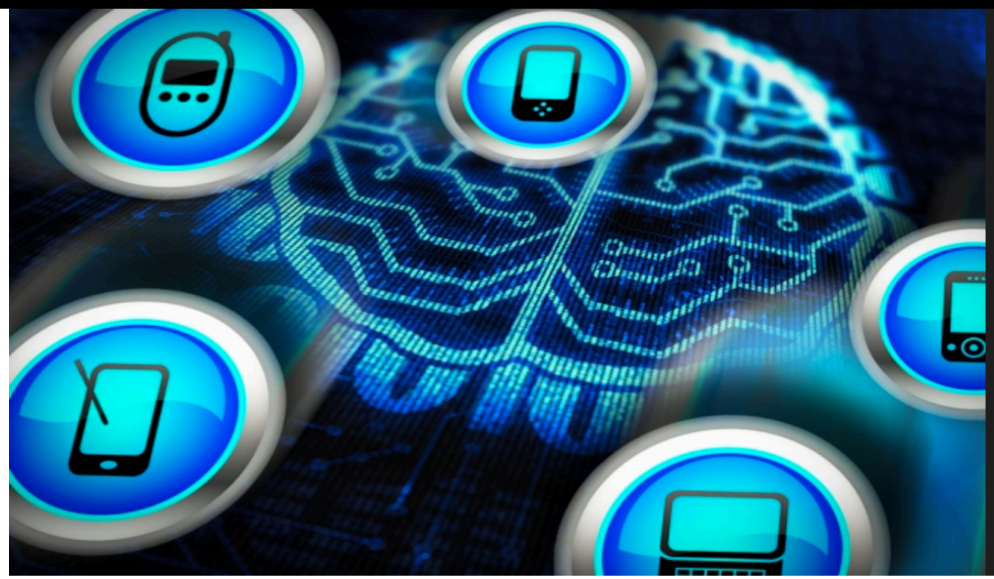
Massachusetts Institute of

NEWS | VIDEO | SOCIAL | FOLLOW MIT

MIT News

ON CAMPUS AND AROUND THE WORLD

Browse or Search



FULL SCREEN

MIT researchers have designed a new chip to implement neural networks. It is 10 times as efficient as a mobile GPU, so it could enable mobile devices to run powerful artificial-intelligence algorithms locally, rather than uploading data to the Internet for processing.

Image: MIT News

Energy-friendly chip can perform powerful artificial-intelligence tasks

Advance could enable mobile devices to implement “neural networks” modeled on the human brain.

Larry Hardesty | MIT News Office
February 3, 2016

Press Inquiries

RELATED

What Kinds of Custom H/W, cont

recode TRENDING TOPICS WRITERS PODCASTS EVENTS

BIG DATA INTEL ARTIFICIAL INTELLIGENCE

Intel is paying more than $400 million to buy deep-learning startup Nervana Systems

The chip giant is betting that machine learning is going to be a big deal in the data center.

BY INA FRIED · AUG 9, 2016, 4:00P

TWEET SHARE LINKEDIN

Intel data center chief Diane Bryant, with Nervana co-founders Naveen Rao, Arjun Bansal, Amir Khosrowshahi and Intel vice president Jason Waxman (left to right) Intel

TRENDING

The Google X moonshot factory is struggling to get products out the door

Bruno Mars and Mark Zuckerberg showed up at Spotify CEO Daniel Ek's

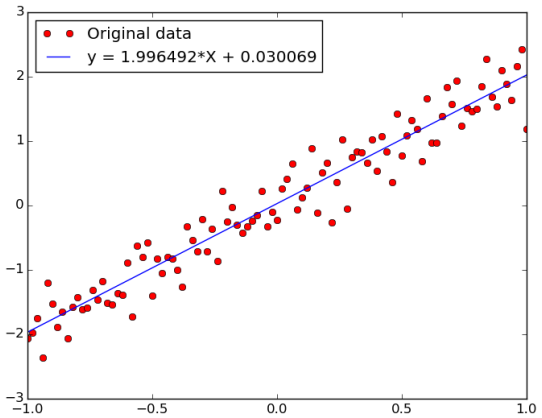
So What Kinds of Network Oriented Use Cases Have We Been Working On?

- Security/Anomaly Detection
- Site Reliability Engineering
- NFV orchestration and optimization
- New automation tools for DevOps
- Predicting and remediating problems mobile networks
- Network control plane optimization
- Network Gamification
- …

Importantly, we can use deep neural networks to capture intuition from experts in these problem domains

Agenda

- What Is Machine Learning?
- Why All The (Machine Learning) Excitement?
- Combining SDN and Machine Learning
- What's On the Horizon?
- Technical explanations/code
  - https://github.com/davidmeyer/ml
  - http://www.1-4-5.net/~dmm/ml



Combining SDN and Machine Learning

Knowledge-Defined Networking

Albert Mestres*, Alberto Rodriguez-Natal*, Josep Carner*, Pere Barlet-Ros*, Eduard Alarcón*,
Marc Solé†, Victor Muntés-Mulero†, David Meyer‡, Sharon Barkai§, Mike J Hibbett¶, Giovani Estrada¶,
Khaldun Ma'ruf‖, Florin Coras**, Vina Ermagan**, Hugo Latapie**, Chris Cassar**, John Evans**, Fabio Maino**,
Jean Walrand†† and Albert Cabellos*
* Universitat Politècnica de Catalunya † CA Technologies ‡ Brocade Communication § Hewlett Packard Enterprise
¶ Intel R&D ‖ NTT Communications ** Cisco Systems †† University of California, Berkeley

***Abstract*—The research community has considered in the past the application of Artificial Intelligence (AI) techniques to control and operate networks. A notable example is the Knowledge Plane proposed by D.Clark et al. However, such techniques have not been extensively prototyped or deployed in the field yet. In this paper, we explore the reasons for the lack of adoption and posit that the rise of two recent paradigms: Software-Defined Networking (SDN) and Network Analytics (NA), will facilitate the adoption of AI techniques in the context of network operation and control. We describe a new paradigm that accommodates and exploits SDN, NA and AI, and provide use cases that illustrate its applicability and benefits. We also present simple experimental results that support its feasibility. We refer to this new paradigm as Knowledge-Defined Networking (KDN).**

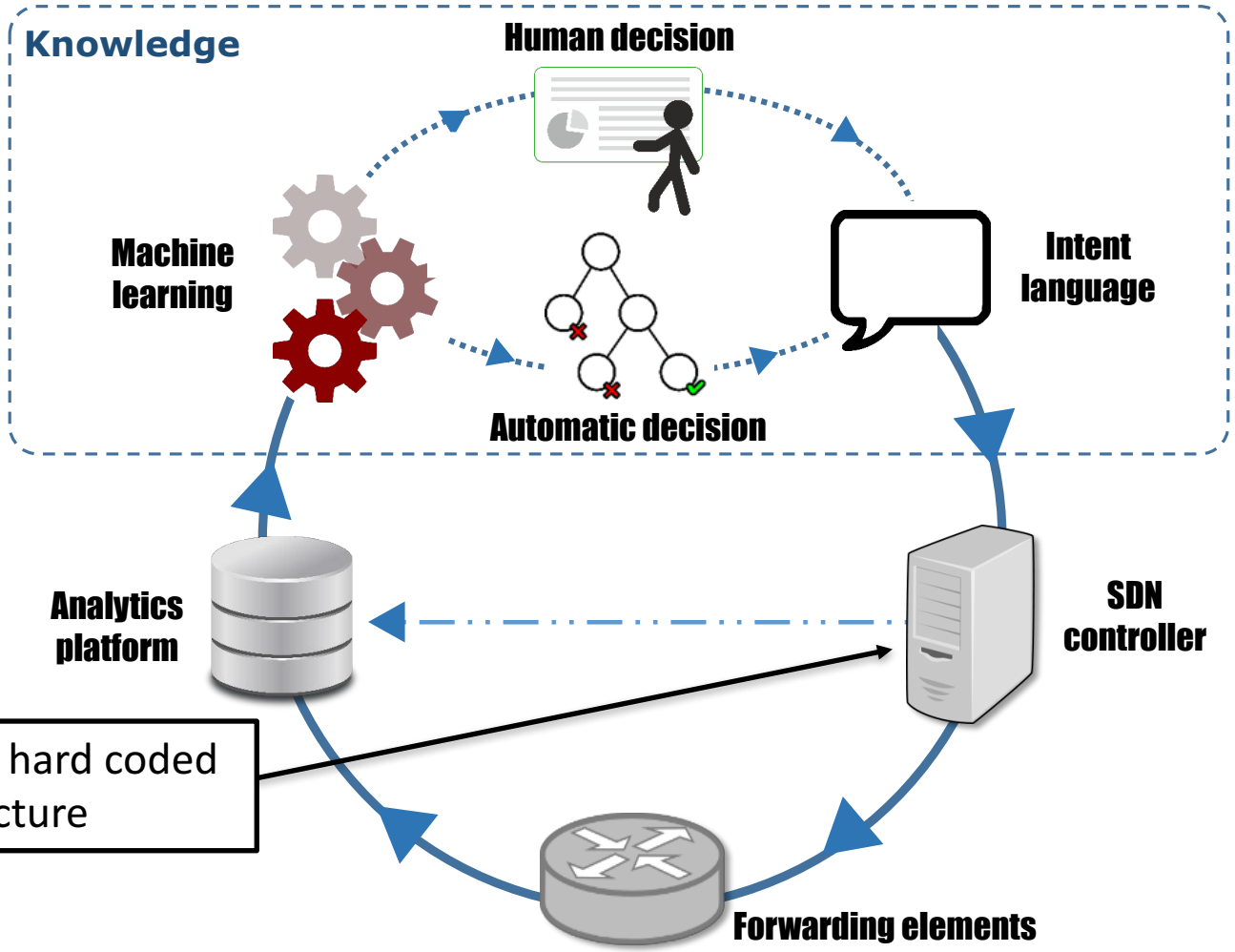
***Keywords*—Knowledge Plane, SDN, Network Analytics, Machine Learning, NFV, Knowledge-Defined Networking**

techniques provide real-time packet and flow-granularity information, as well as configuration and network state monitoring data, to a centralized Network Analytics (NA) platform [4]. In this context, telemetry and analytics technologies provide a richer view of the network compared to what was possible with conventional network management approaches.

In this paper, we advocate that the centralized control offered by SDN, combined with a rich centralized view of the network provided by network analytics, enable the deployment of the KP concept proposed in [1]. In this context, the KP can use ML and Deep Learning (DL) techniques to gather knowledge about the network, and exploit that knowledge to control the network using logically centralized control capabilities provided by SDN. We refer to the paradigm resulting from combining SDN, telemetry, Network Analytics, and the Knowledge Plane as Knowledge-Defined Networking.

https://arxiv.org/abs/1606.06222

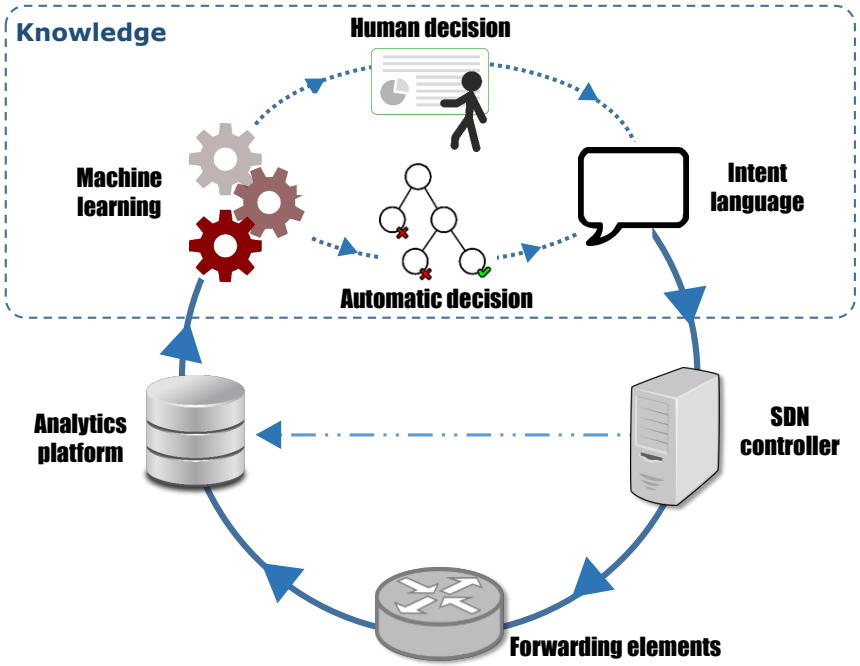
Knowledge-Defined Networking
Basic Control Loop



Note: There is hard coded logic in this picture

Graphic courtesy Albert Cabellos

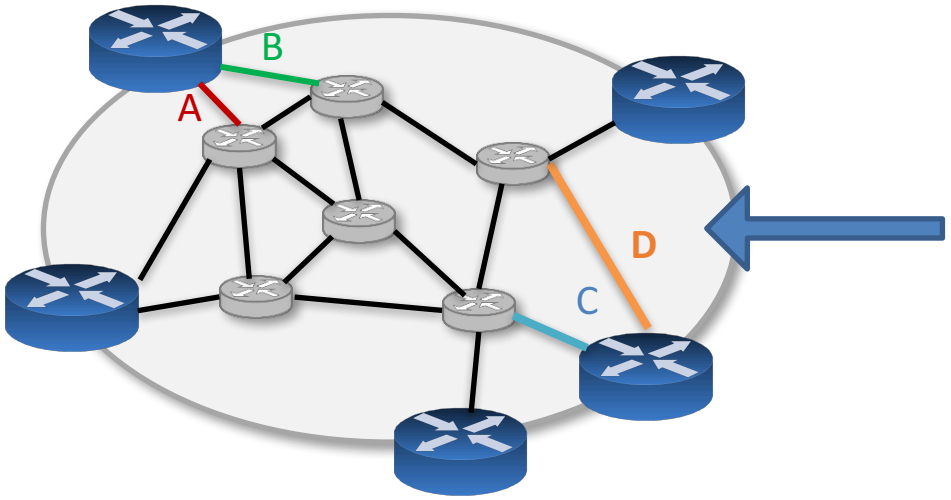
Kinds of Applications We Are Doing?



- Recommendation
- Optimization
  - Hidden Information
  - Complex systems
- Estimation
  - Performance/Cost
- Validation
  - Performance/Cost
- Knowledge discovery

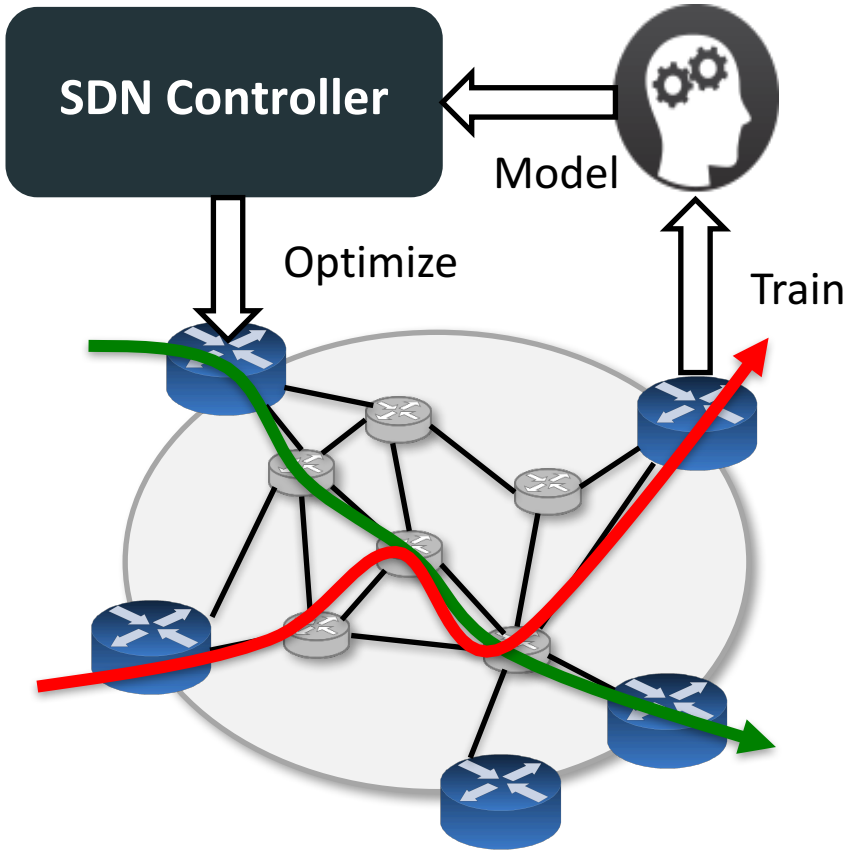
Graphic courtesy Albert Cabellos

Example: Can We Learn Optimal Paths In An Overlay Network?



- Which egress/ingress links should overlay routers use? E.g. **A** or **B** **and** **C** **or** **D**?
  - Underlay is assumed that has an arbitrary constant routing
  - Underlay is assumed as hidden and not under our control
  - Overlay protocol is assumed to be able to choose egress and ingress links, we refer to this as routing policy
- **Goal: Achieve overall minimum latency between overlay nodes**

Optimizing Paths, cont

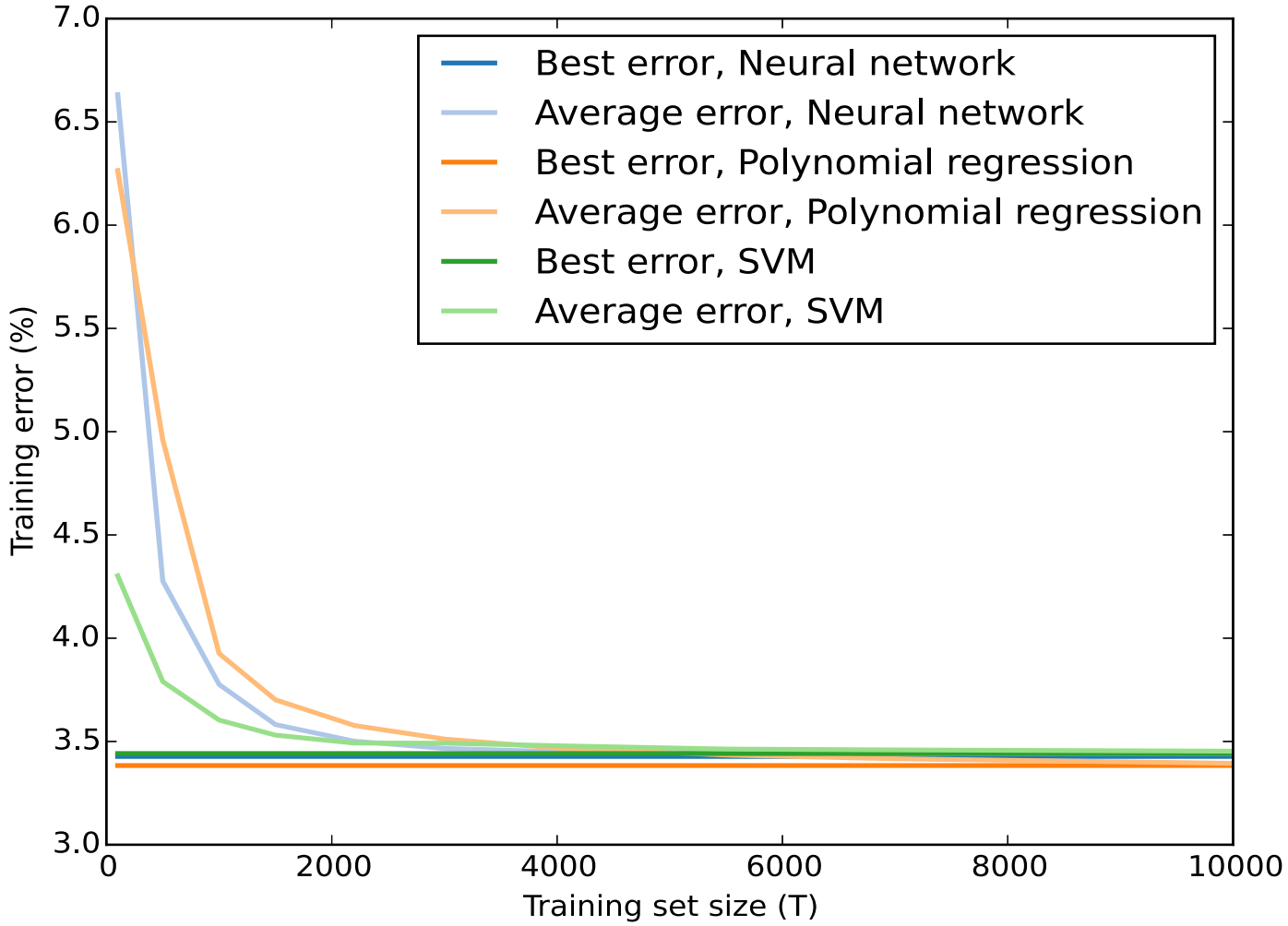


- What we want to predict
  - For a given traffic matrix, predict the ingress/egress link configuration that minimizes delay between ingress and egress
- Training Set
  - $\{(x^{(i)},y^{(i)})\}$, i = 1,…,N
  - $x^{(i)} = (src^{(i)}, dst^{(i)}, bandwitdth^{(i)})$
  - $y^{(i)} = delay^{(i)}$
  - Loss function: sum of delays (minimize this)
  - Learn f: X → Y
  - $y^{(i)} = f(x^{(i)})$
  - For a new **x** predict **y**
- Models
  - Neural Network
  - SVM
  - Polynomial Regession

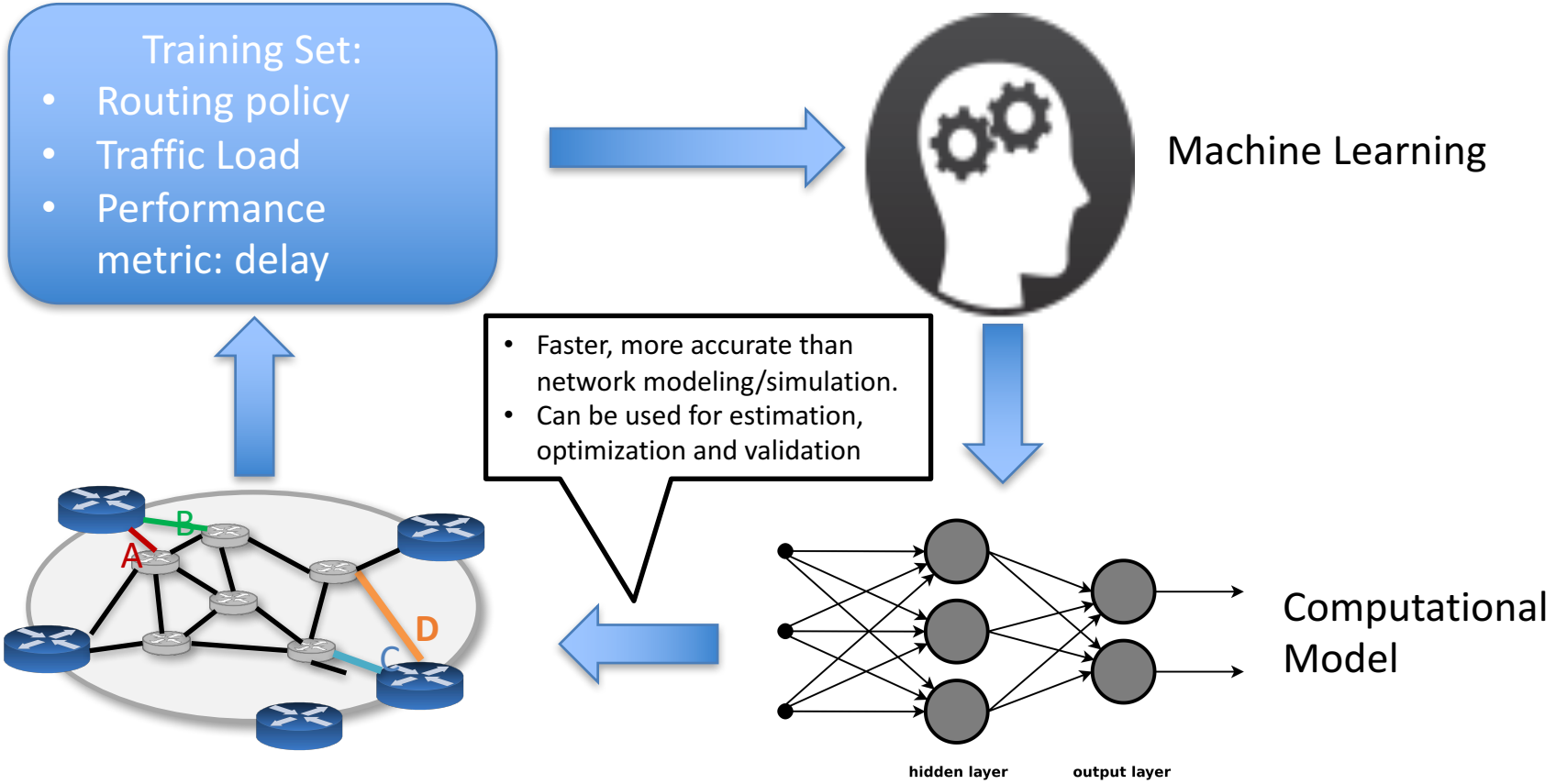
http://knowledgedefinednetworking.org/

Simulation: MSE vs. Training Set Size

(Scale-free, Poisson traffic, 9 Active stations)



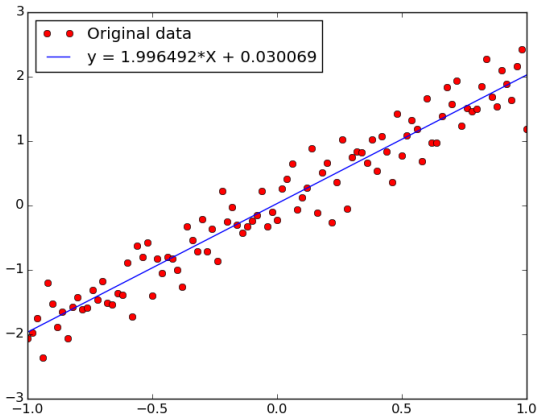
System View



Graphic courtesy Albert Cabellos

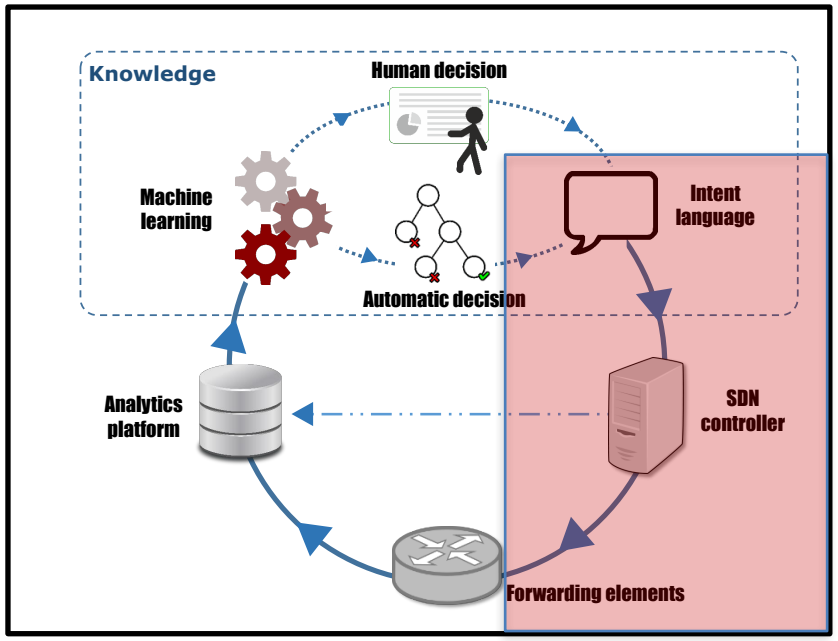
Agenda

- What Is Machine Learning?
- Why All The (Machine Learning) Excitement?
- Combining SDN and Machine Learning
- What's On the Horizon?
- Technical explanations/code
  - https://github.com/davidmeyer/ml
  - http://www.1-4-5.net/~dmm/ml

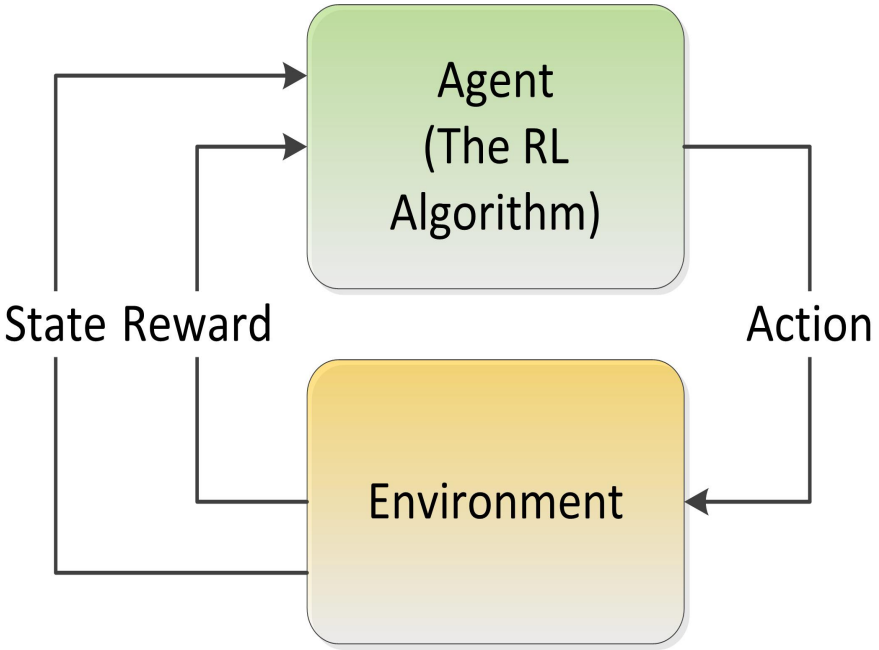


But There's More: We Can Learn Control

Reinforcement Learning Meets Monte Carlo Tree Search and Deep Learning



Hard coded control logic

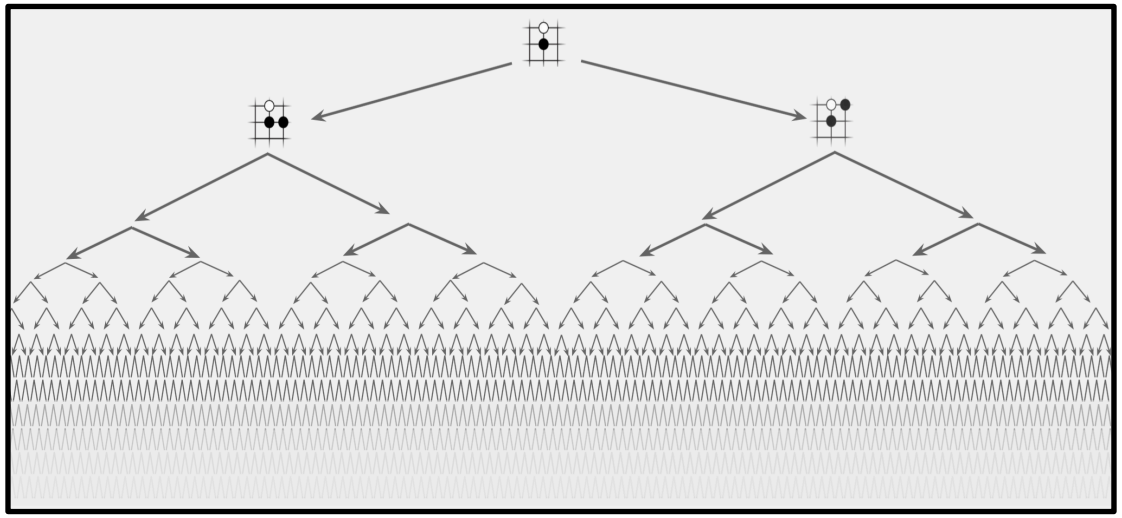


- Security: Agent can learn dynamic/evolving behavior of adversary
- DevOPs: Agent can learn workflow automation (e.g. Openstack Mistral)
- Orchestration: Agent can learn dynamic behavior of VNFs/system
- Deep policy net, $\pi(s) = a, s \in S, a \in A(s)$, can capture human intuition

http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html

BTW, Why Is *Go* So Hard?

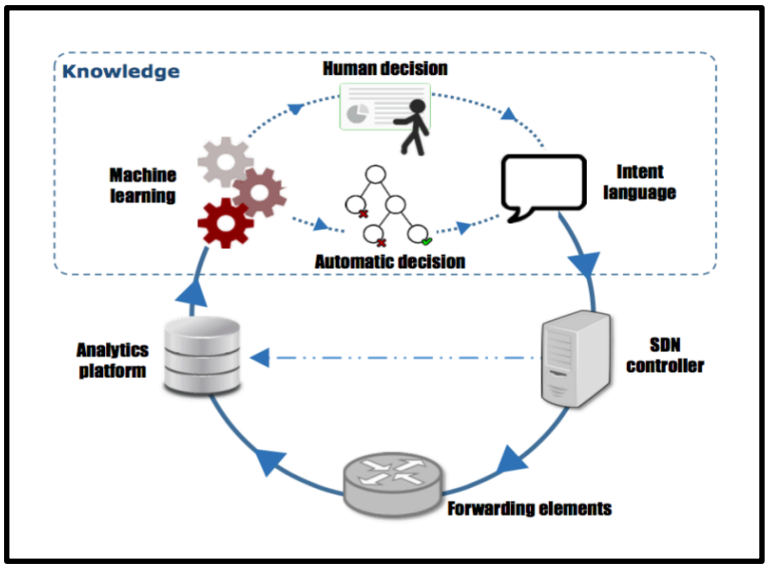
- Game Tree Complexity = b^d



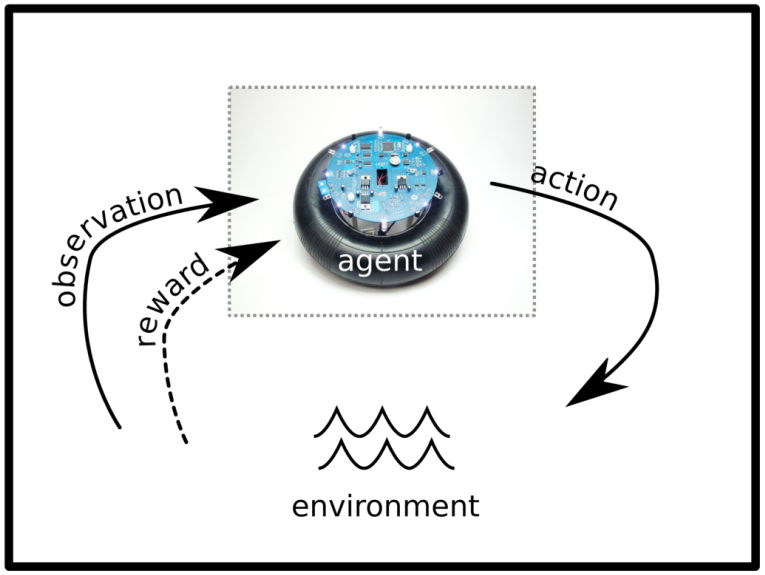
- Brute Force Search Intractable
  - Search space is huge (10^721)
  - Difficult to evaluate who is winning

- Can we characterize network state and action spaces?
  - And why is this important?

Static vs. Reinforcement Learning Architectures



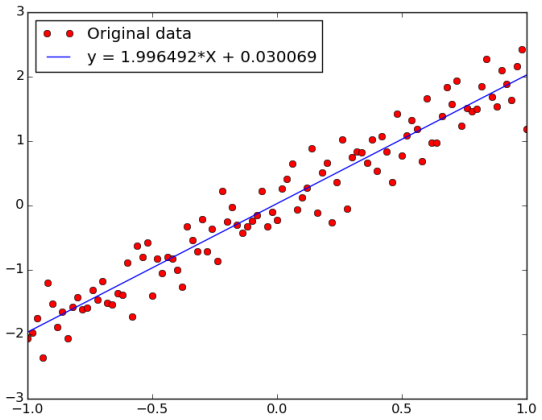
- Today's Static ML Architectures
  - ML doesn't have "agency"
  - Hard-coded or open loop control
- Doesn't learn control
- Assumes stationary DGD
- Largely off-line



- Reinforcement Learning Architecture
  - Agent architecture
  - Agent learns control/action selection
- Adapts to evolving environment
- Non-stationary distributions
- *Network gamification*

Agenda

- What Is Machine Learning?
- Why All The (Machine Learning) Excitement?
- Combining SDN and Machine Learning
- What's On the Horizon?
- Technical explanations/code
  - https://github.com/davidmeyer/ml
  - http://www.1-4-5.net/~dmm/ml



Summary

- Lots of Network Data
  - Standardized and labeled datasets still scarce
- Lots of open source ML frameworks and examples
  - Tensorflow (tensorflow.org)
  - Torch (torch.ch)
  - Scikit-learn (scikit-learn.org)
  - …
- Still requires skill/experience to
  - Build DNN architectures
  - Find ”good” settings for hyper-parameters
  - Prevent overfitting
  - …

Company¤	Cloud·Machine·Learning·Platform¤	Deep·Learning·Now·Open·Source¤
Amazon¤	Amazon·Machine·Learning¤	DSSTNE·(sounds·like·Destiny)·Deep·Scalable·Sparse·Tensor·Network·Engine¤
Baidu¤	None¤	Deep·Speech·2¤
Facebook¤	None¤	TorchNet·(built·on·the·previously·open·source·Torch·library)¤
Google¤	Google·NEXT·Cloud·Platform¤	TensorFlow¤
IBM¤	IBM·Watson·Analytics¤	IBM·SystemML¤
Microsoft¤	Azure·Machine·Learning¤	CNTK—Computational·Network·Toolkit¤
Twitter¤	None¤	Twitter·Cortex¤
Yahoo¤	None¤	CaffeOnSpark¤

- You will be seeing Machine Learning in all facets of Networking
  - And everything else for that matter
- All of that said, the time to get your feet wet is now!

Q&A

Thank you

(have more questions/comments? dmm@1-4-5.net)