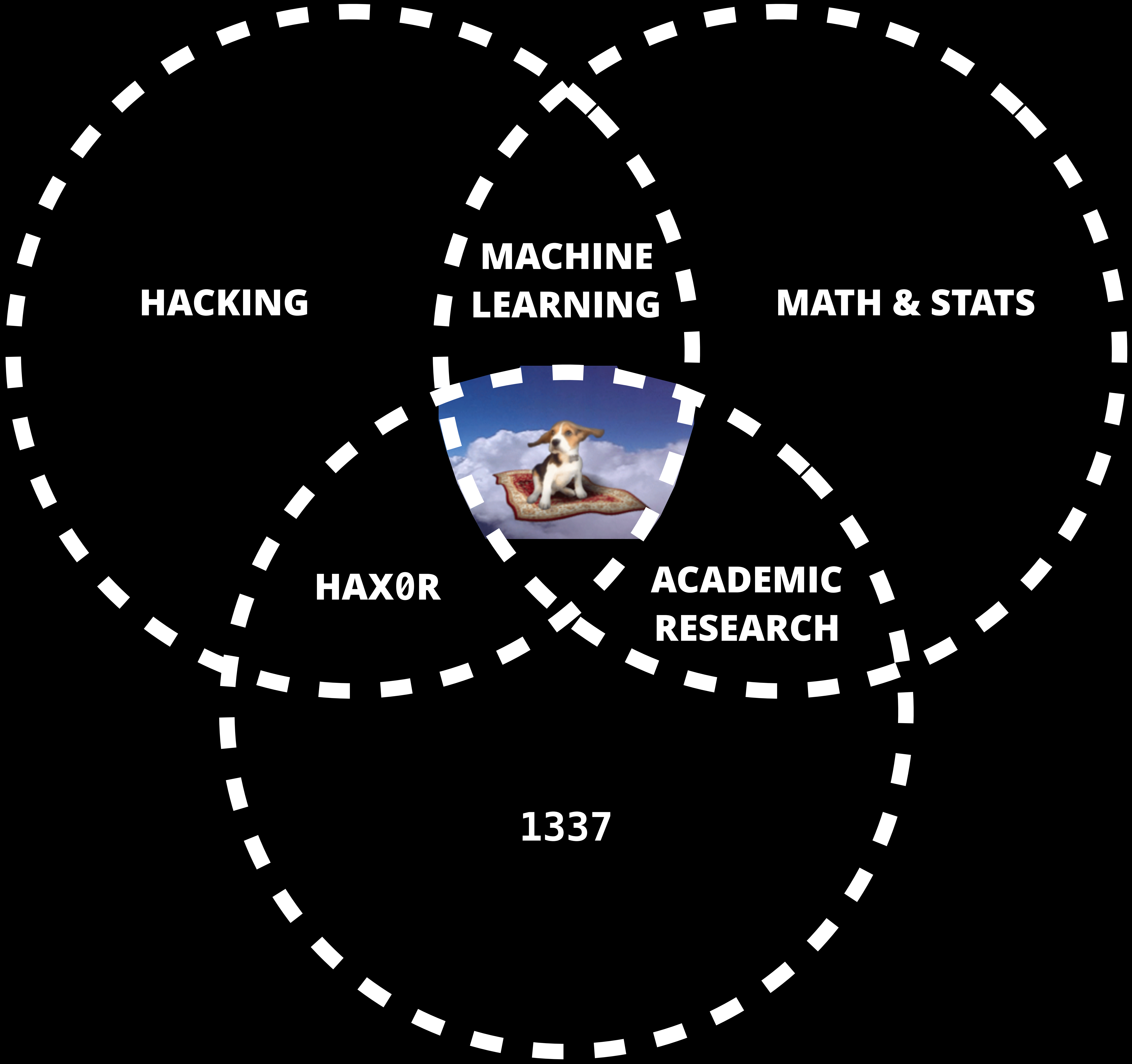
Machine Duping
Pwning Deep Learning Systems

CLARENCE CHIO
MLHACKER
@cchio

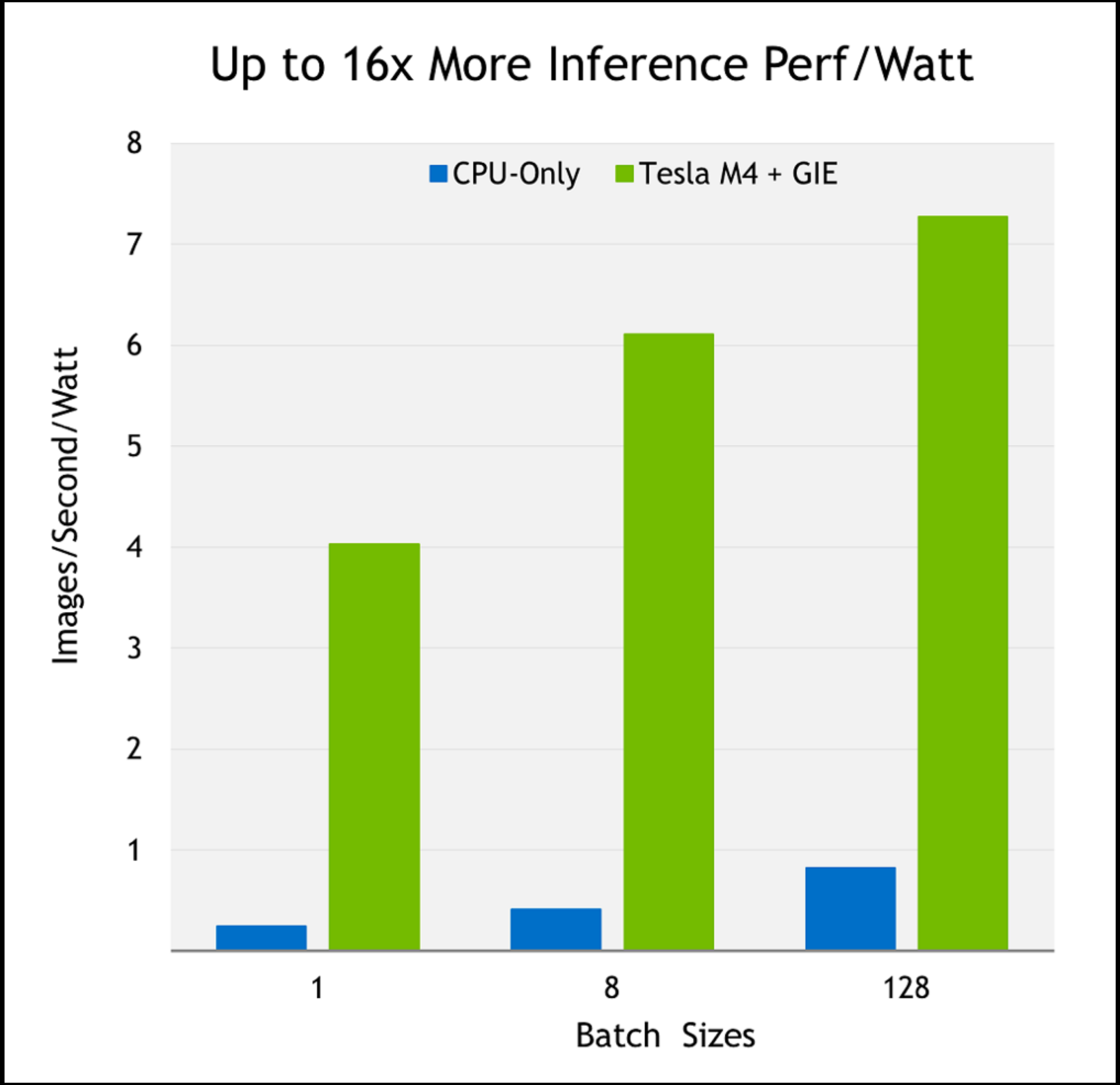


adapted from
Dave Conway

Deep Learning

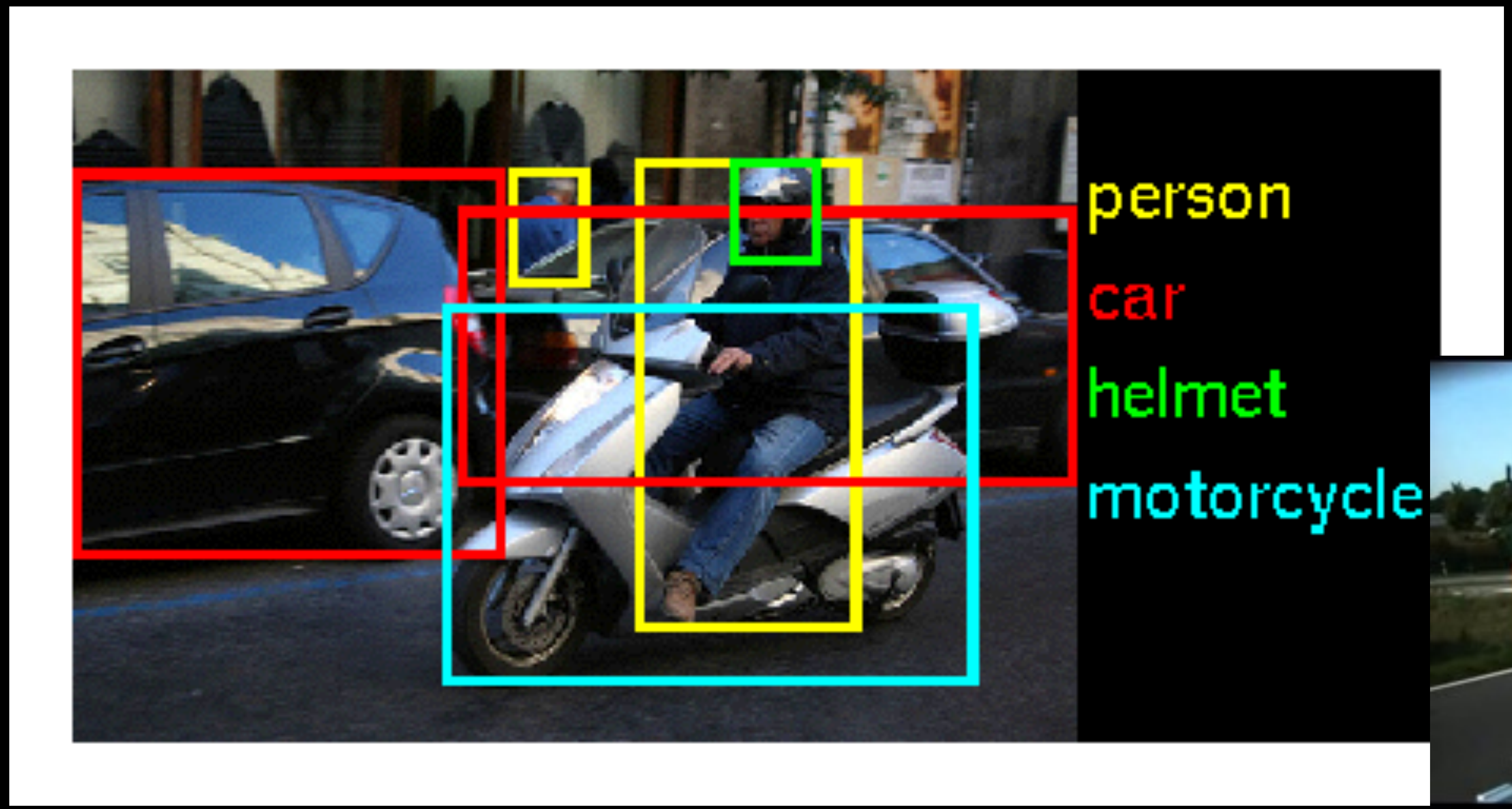
deep neural networks (DNN)

- Not a new toy - history goes back to **1943**
- MUCH MORE **DATA** EVERYWHERE
- Revived due to improvements in computational hardware (esp. GPUs)
    - Multiple concurrent matrix operations can be performed
- MYTH: “Modeled after how the human brain works”

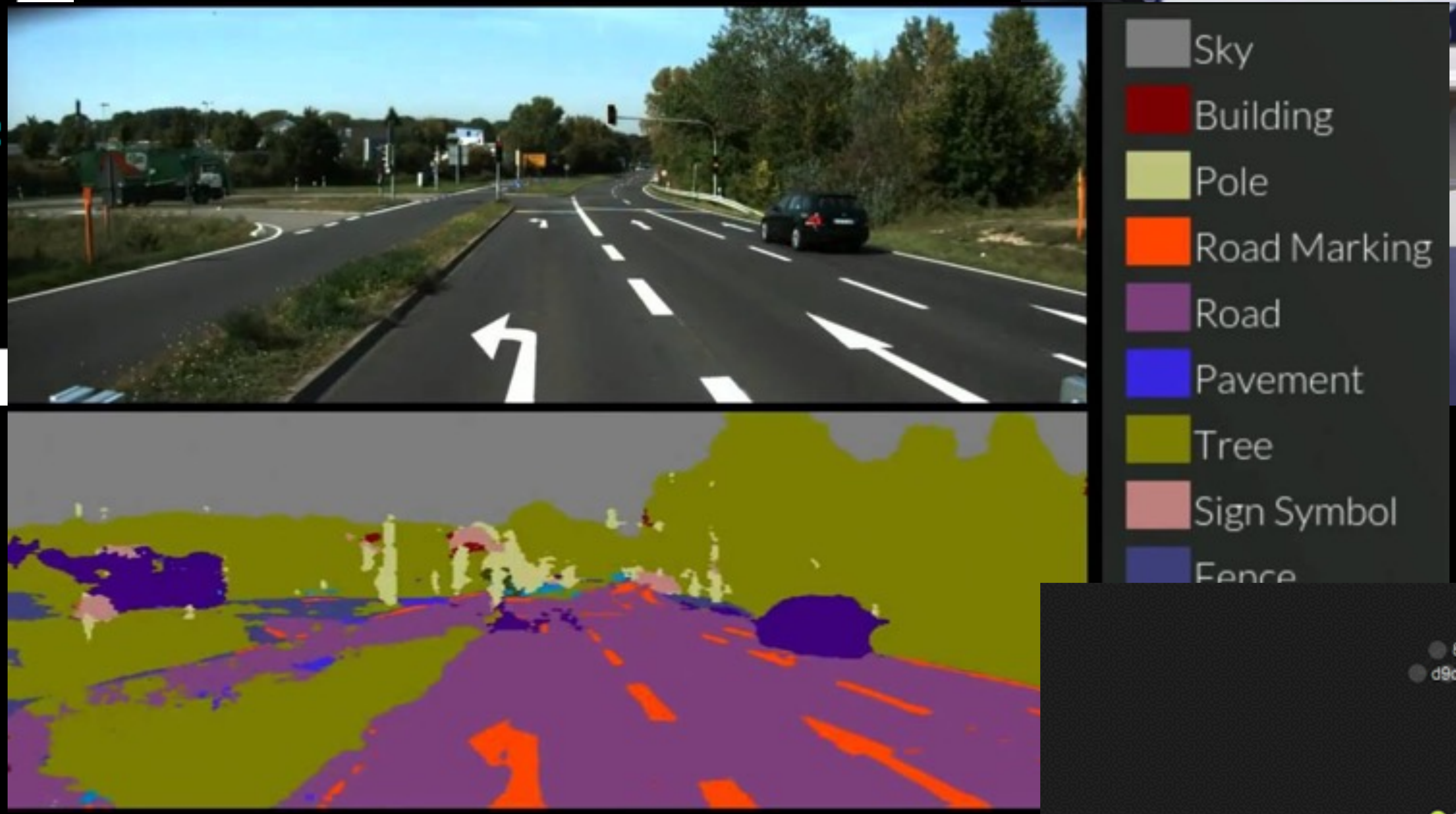


Up to 16x More Inference Perf/Watt
CPU-Only
Tesla M4 + GIE
Images/Second/Watt
8
7
6
5
4
3
2
1
1
8
128
Batch Sizes

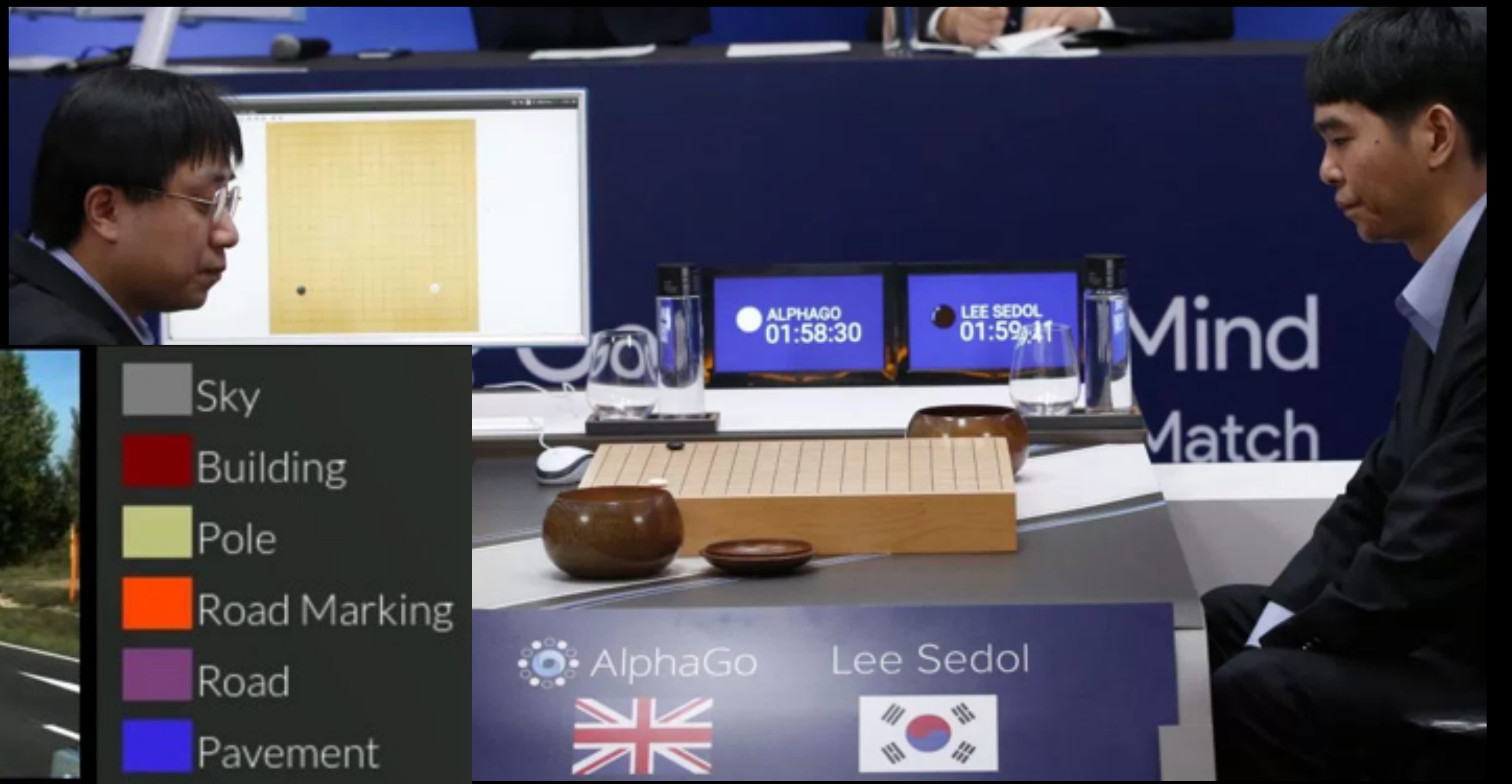
Nvidia



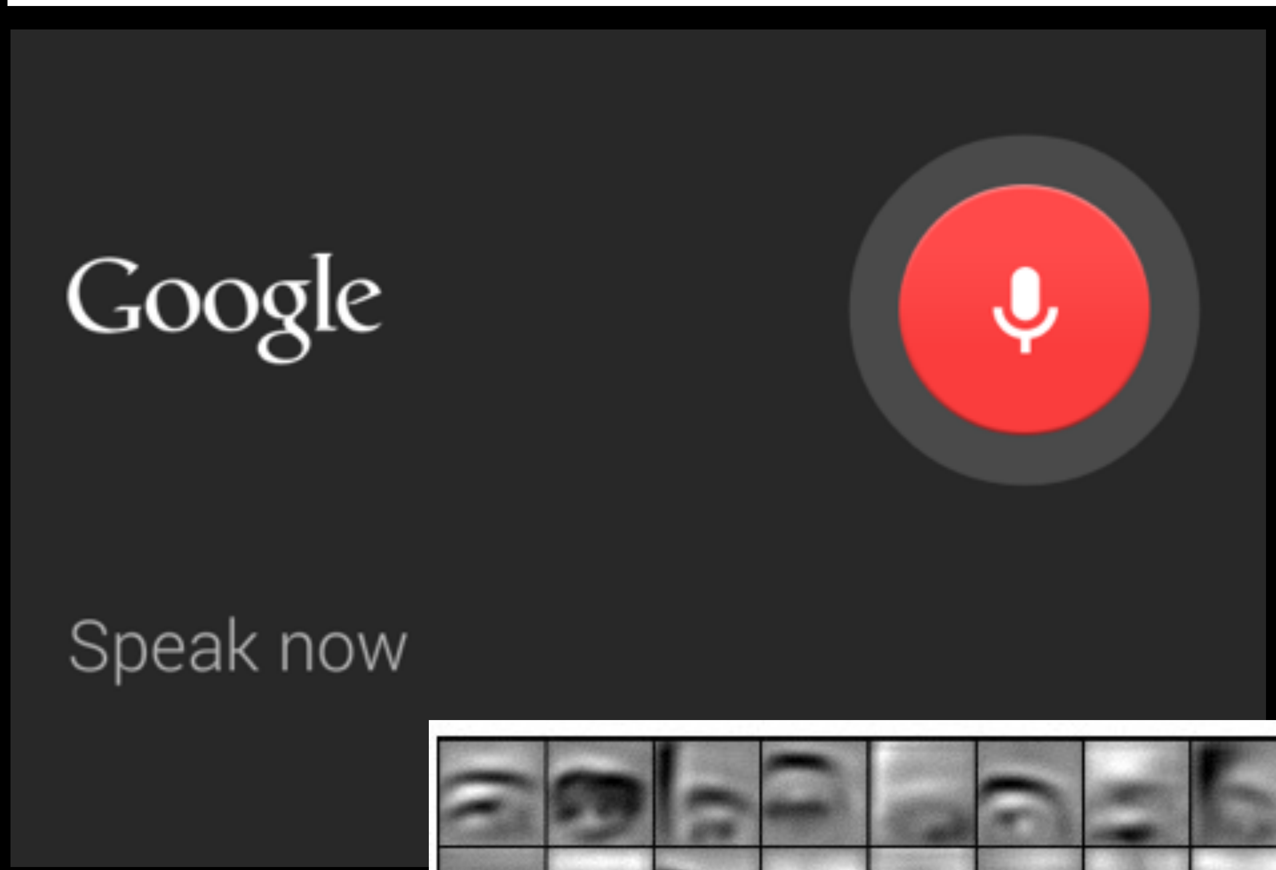
ImageNet



Nvidia DRIVE

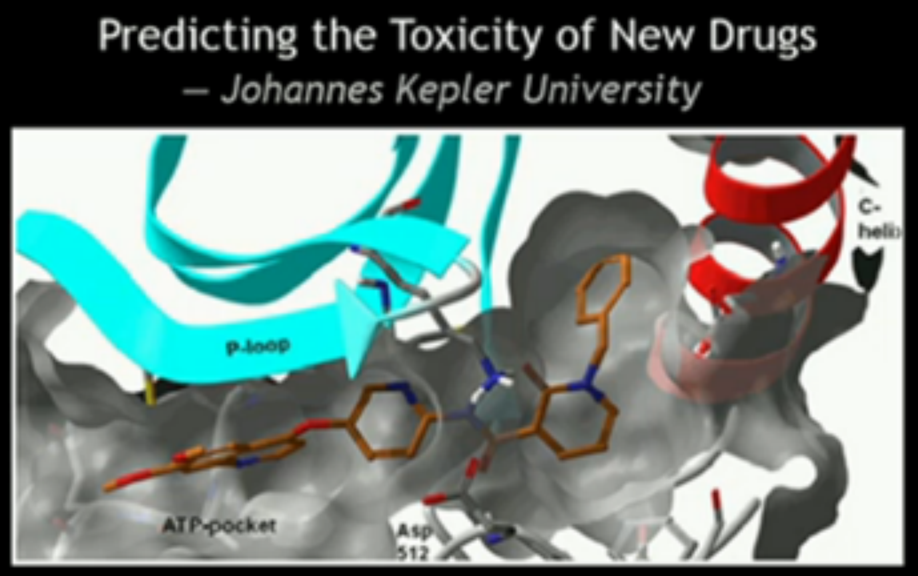


Google DeepMind

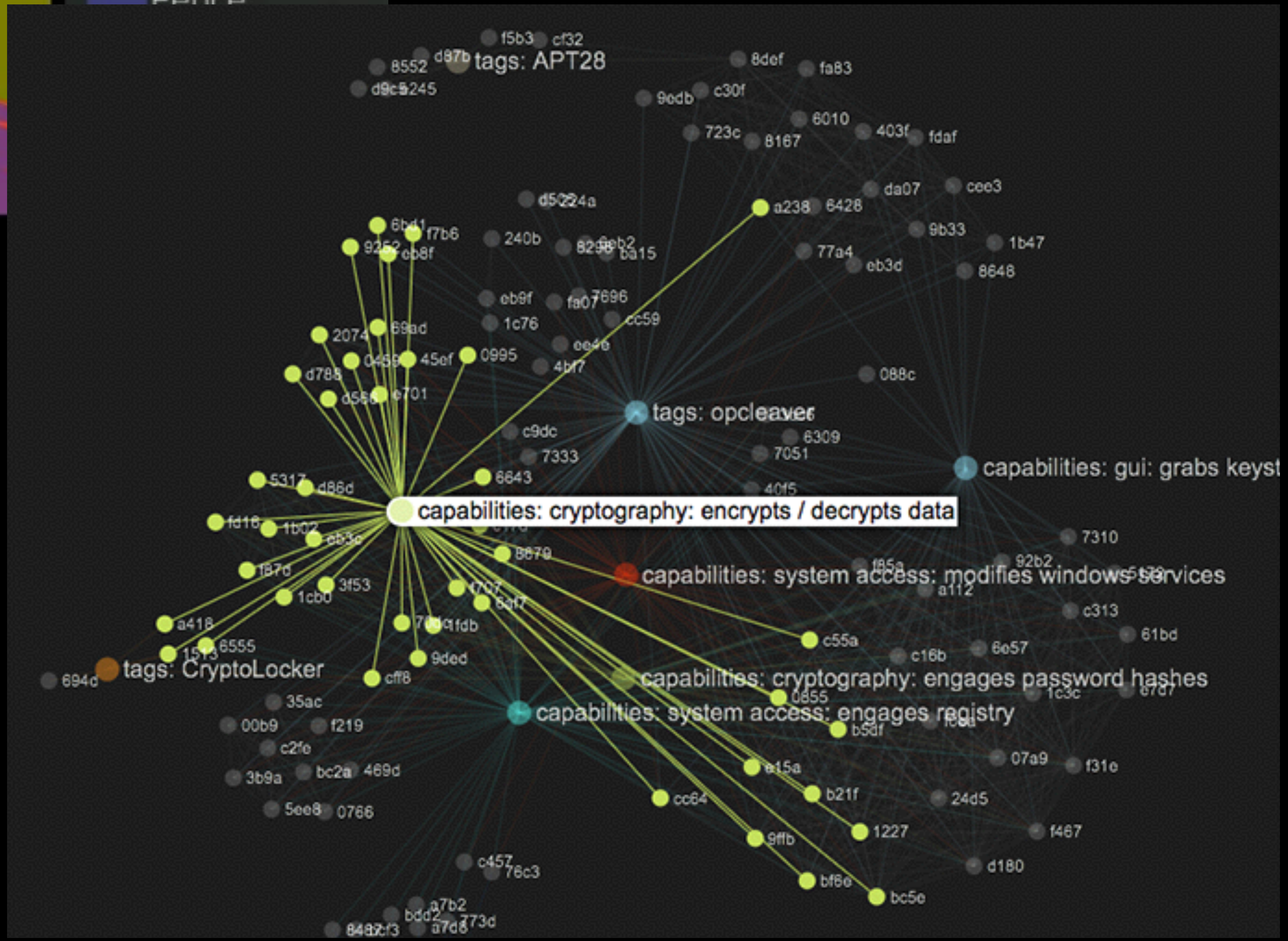


Google Inc.

UMass, Facial Recognition



Nvidia



Invincea Malware Detection System

Deep Learning

why would someone choose to use it?

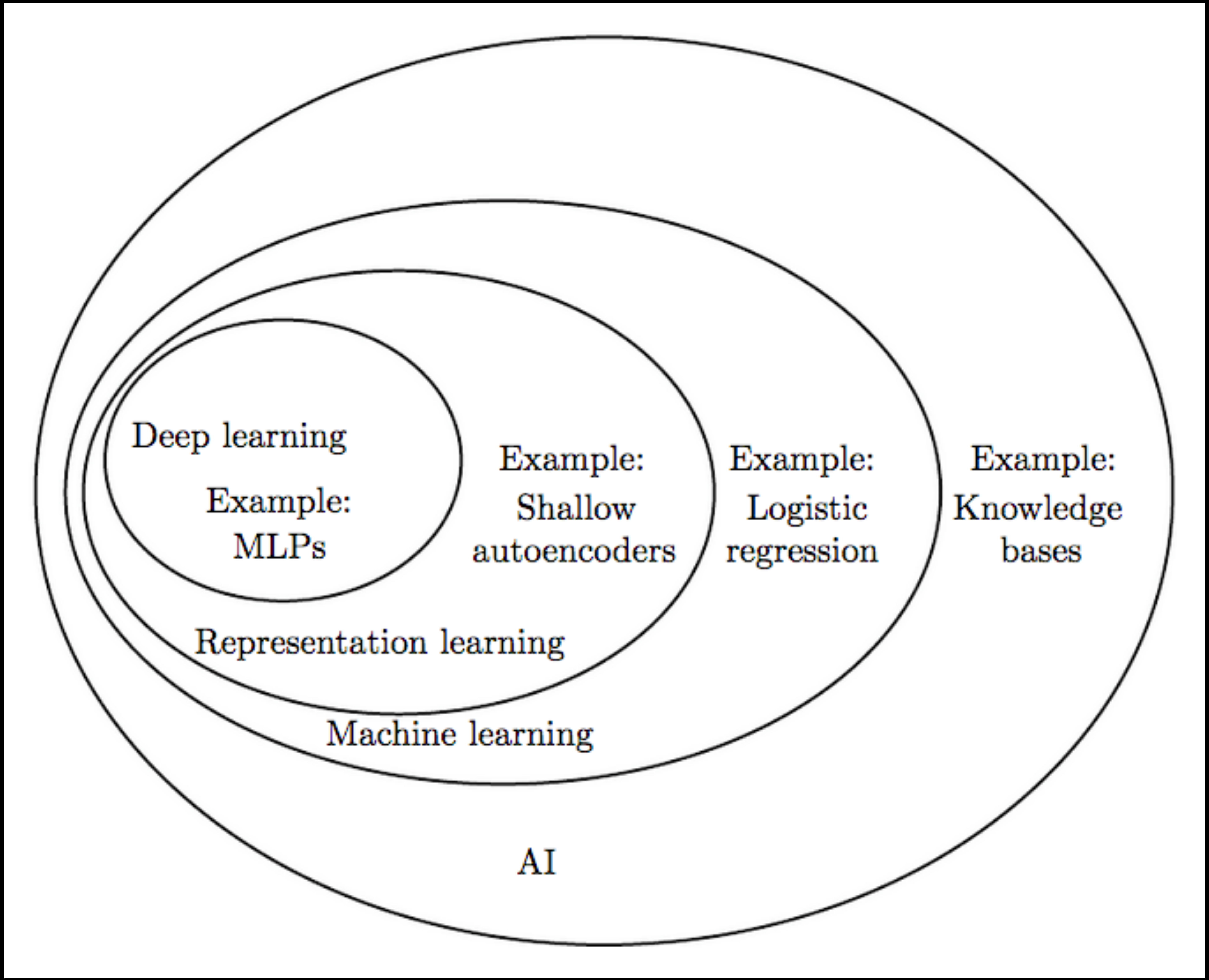
(Semi)Automated feature/representation learning

One infrastructure for multiple problems (sort of)

Hierarchical learning: Divide task across multiple layers

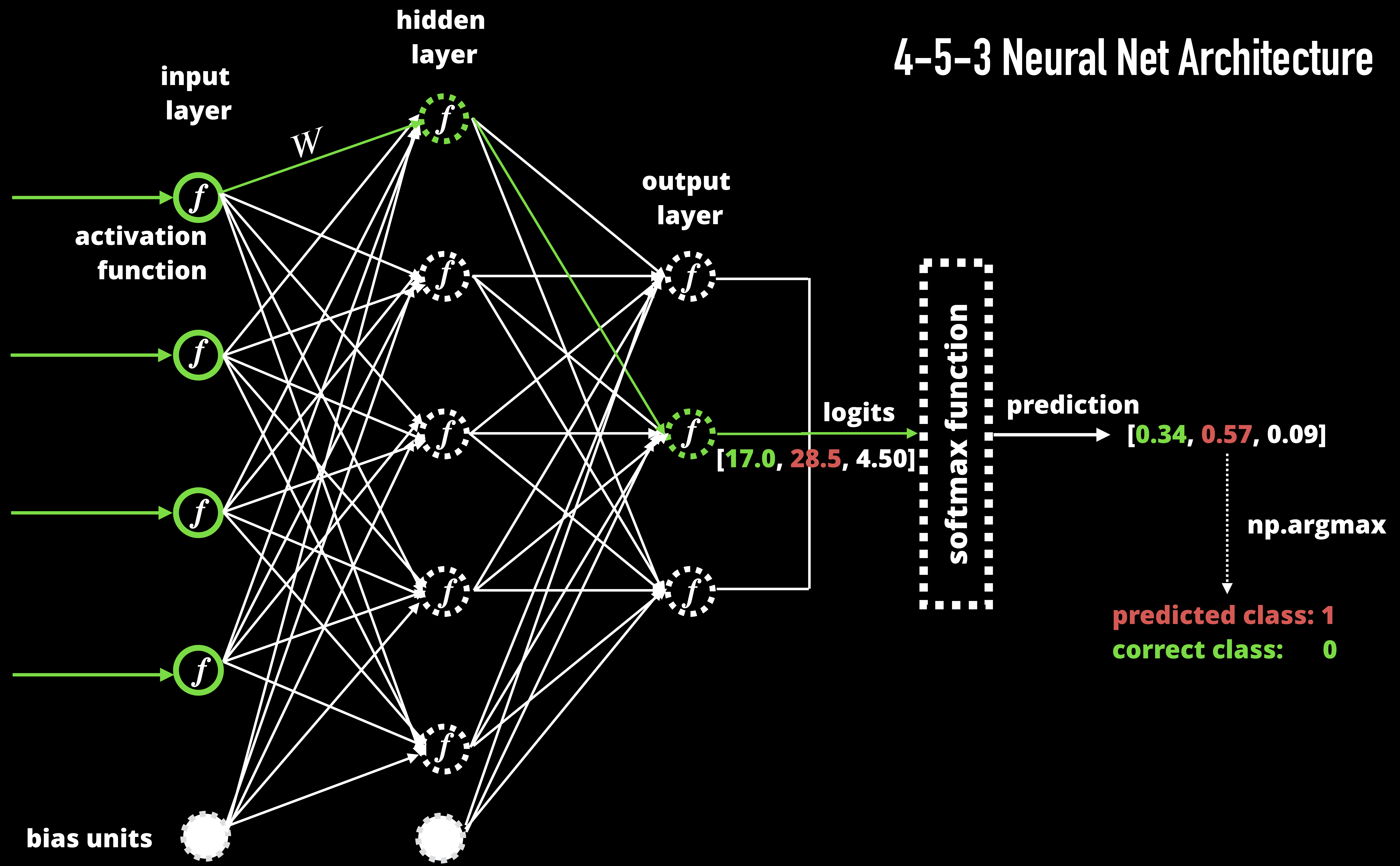
Efficient, easily distributed & parallelized

Definitely not one-size-fits-all



Source: Deep Learning: Goodfellow et. al.

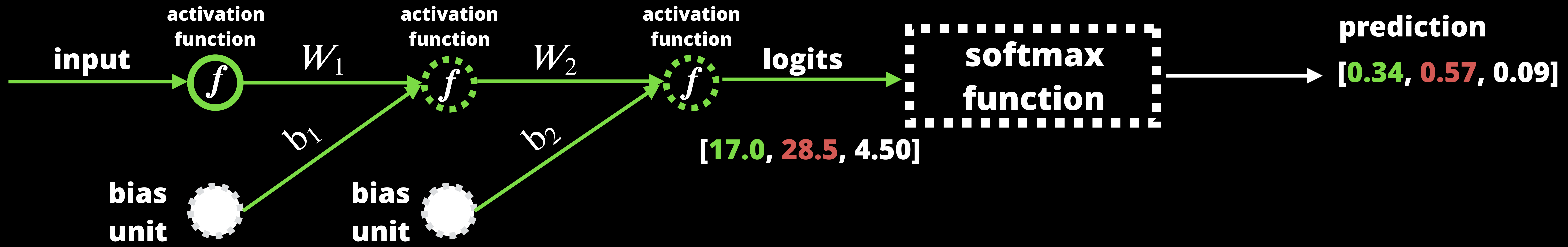
4-5-3 Neural Net Architecture



Training Deep Neural Networks

Step 1 of 2: Feed Forward

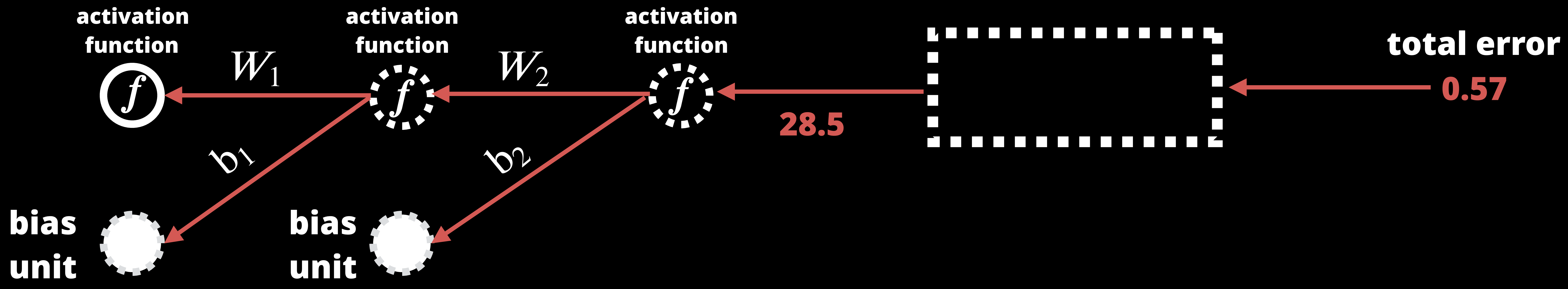
1. Each unit receives output of the neurons in the previous layer (+ bias signal)
2. Computes weighted average of inputs
3. Apply weighted average through nonlinear activation function
4. For DNN classifier, send final layer output through softmax function



Training Deep Neural Networks

Step 2 of 2: Backpropagation

1. **If the model made a wrong prediction, calculate the error**
   1. In this case, the correct class is 0, but the model predicted 1 with 57% confidence - error is thus 0.57
2. **Assign blame: trace backwards to find the units that contributed to this wrong prediction (and how much they contributed to the total error)**
   1. Partial differentiation of this error w.r.t. the unit's activation value
3. **Penalize those units by decreasing their weights and biases by an amount proportional to their error contribution**
4. **Do the above efficiently with optimization algorithm e.g. Stochastic Gradient Descent**

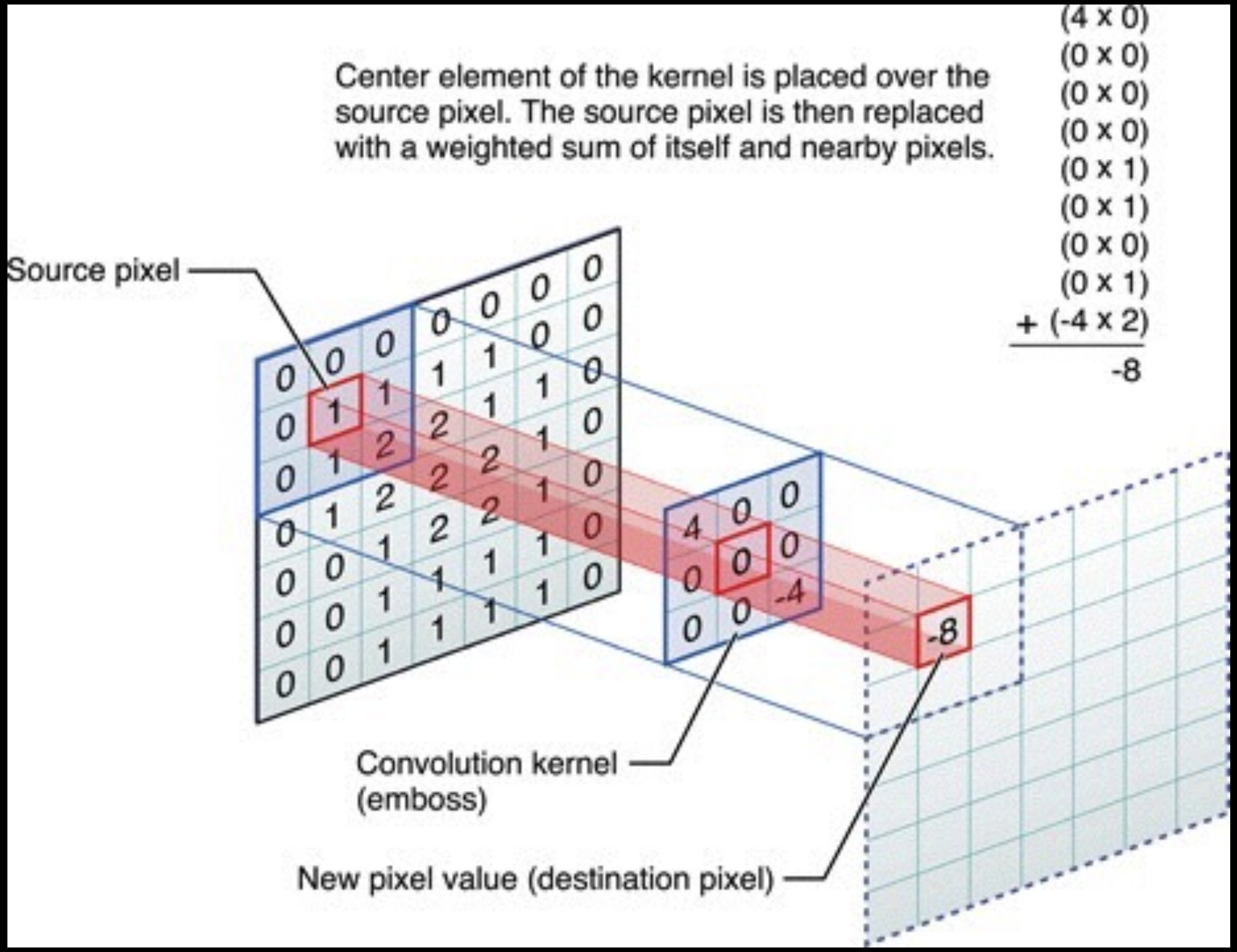


HOW?

DEMO

Beyond Multi Layer Perceptrons

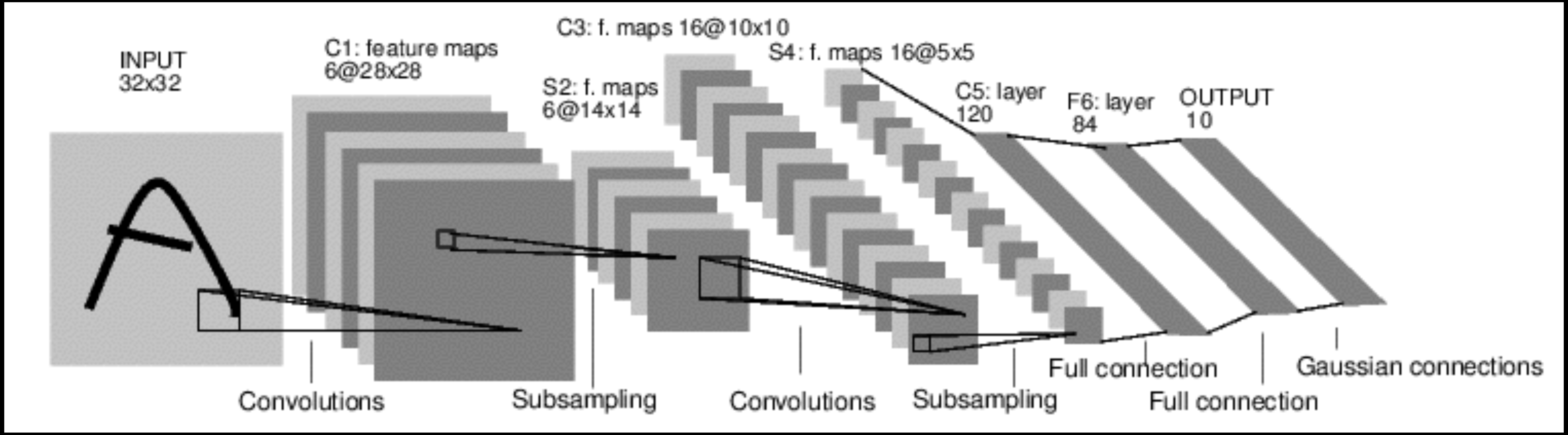
Convolutional Neural Network



Source: iOS Developer Library
vImage Programming Guide

Beyond Multi Layer Perceptrons

Convolutional Neural Network

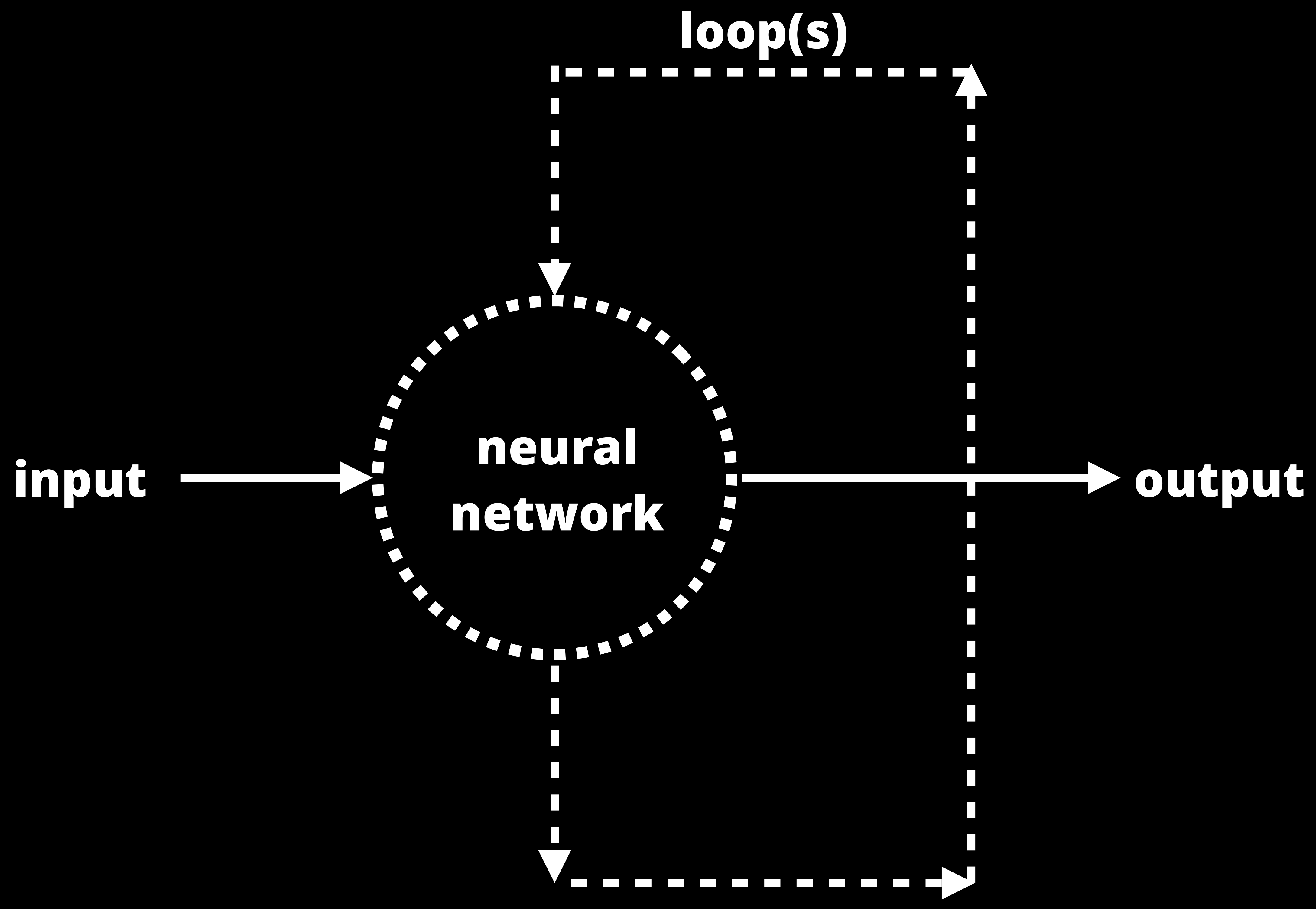


Source: LeNet 5, LeCun et. al.

Beyond Multi Layer Perceptrons

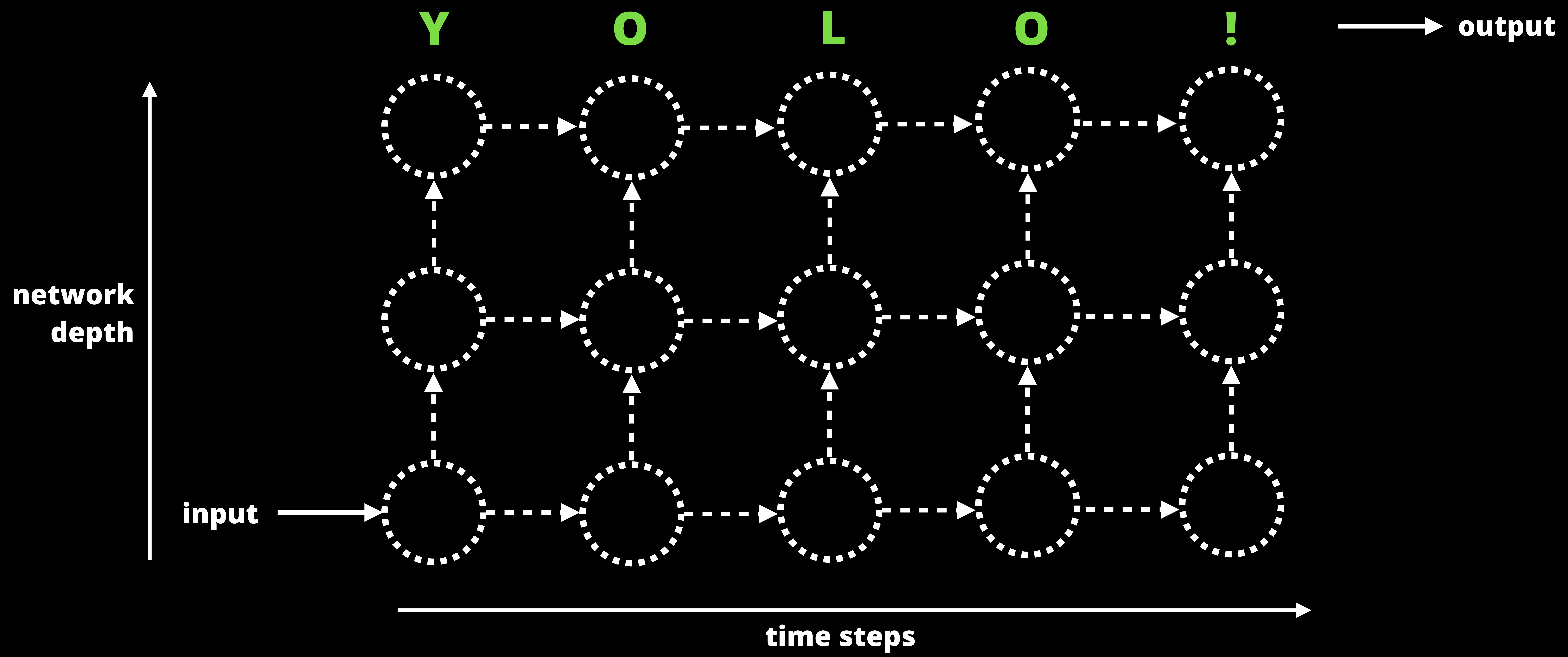
Recurrent Neural Network

- Just a DNN with a feedback loop
- Previous time step feeds all intermediate and final values into next time step
- Introduces the concept of “memory” to neural networks



Beyond Multi Layer Perceptrons

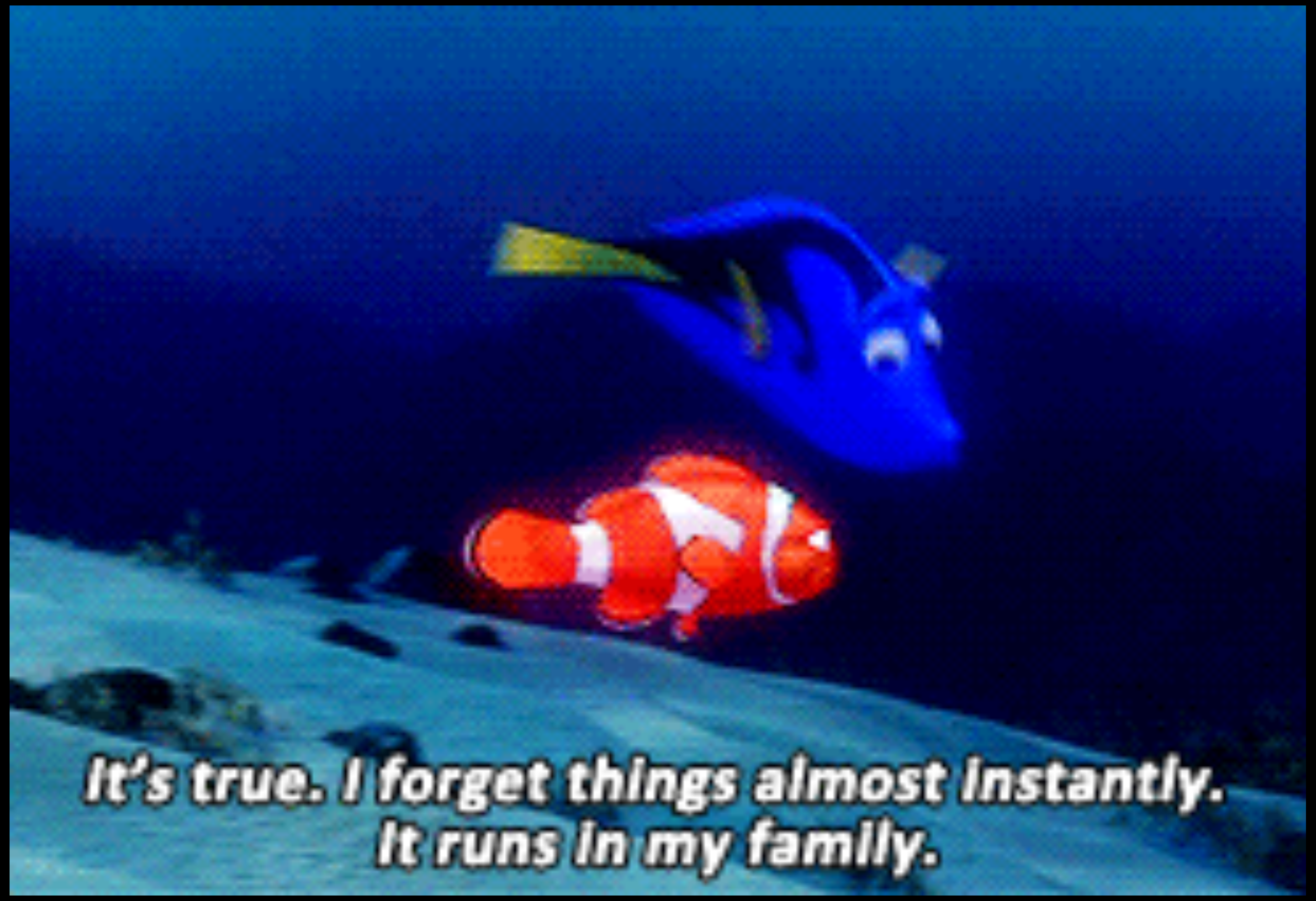
Recurrent Neural Network



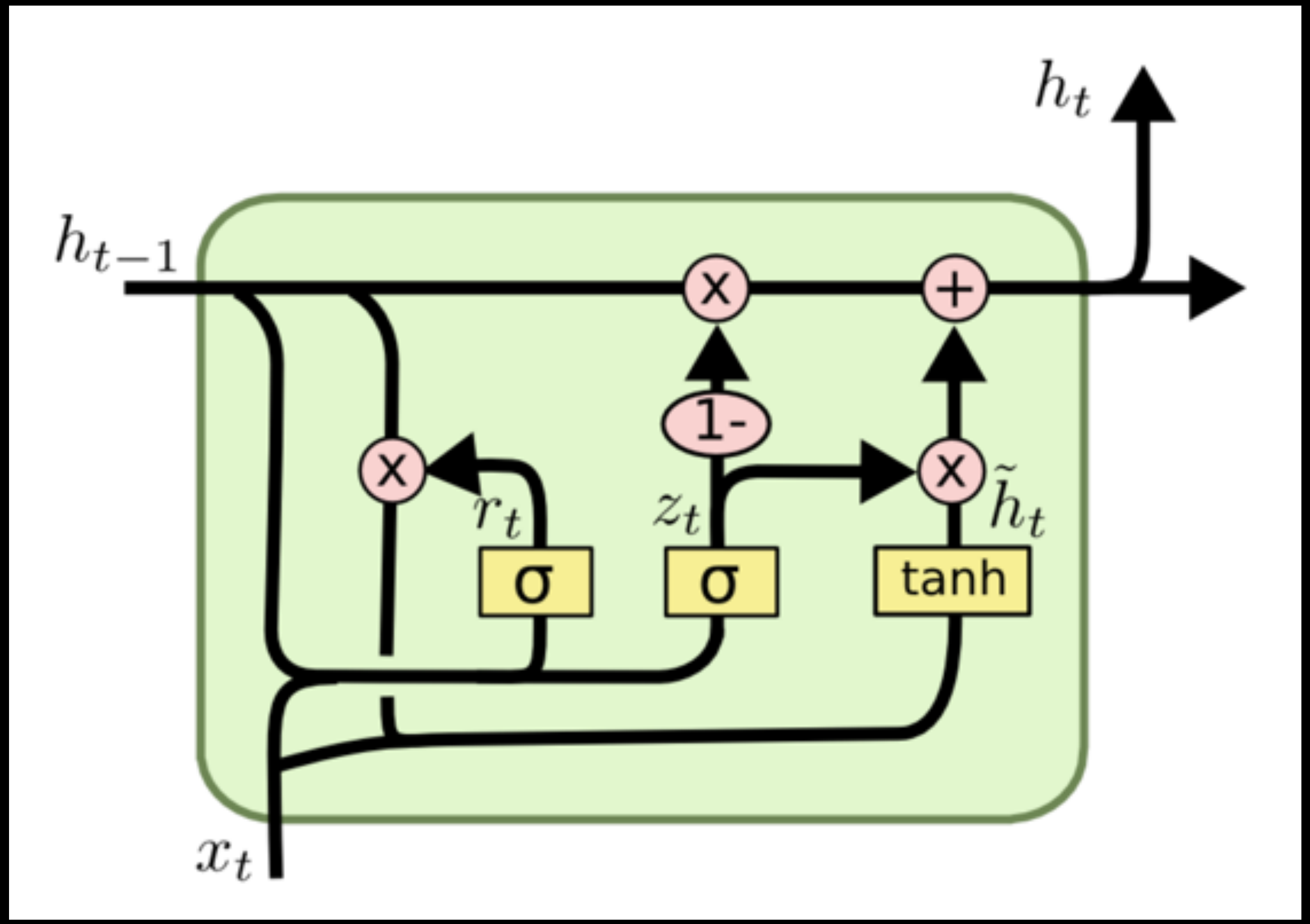
Beyond Multi Layer Perceptrons

Long Short-Term Memory (LSTM) RNN

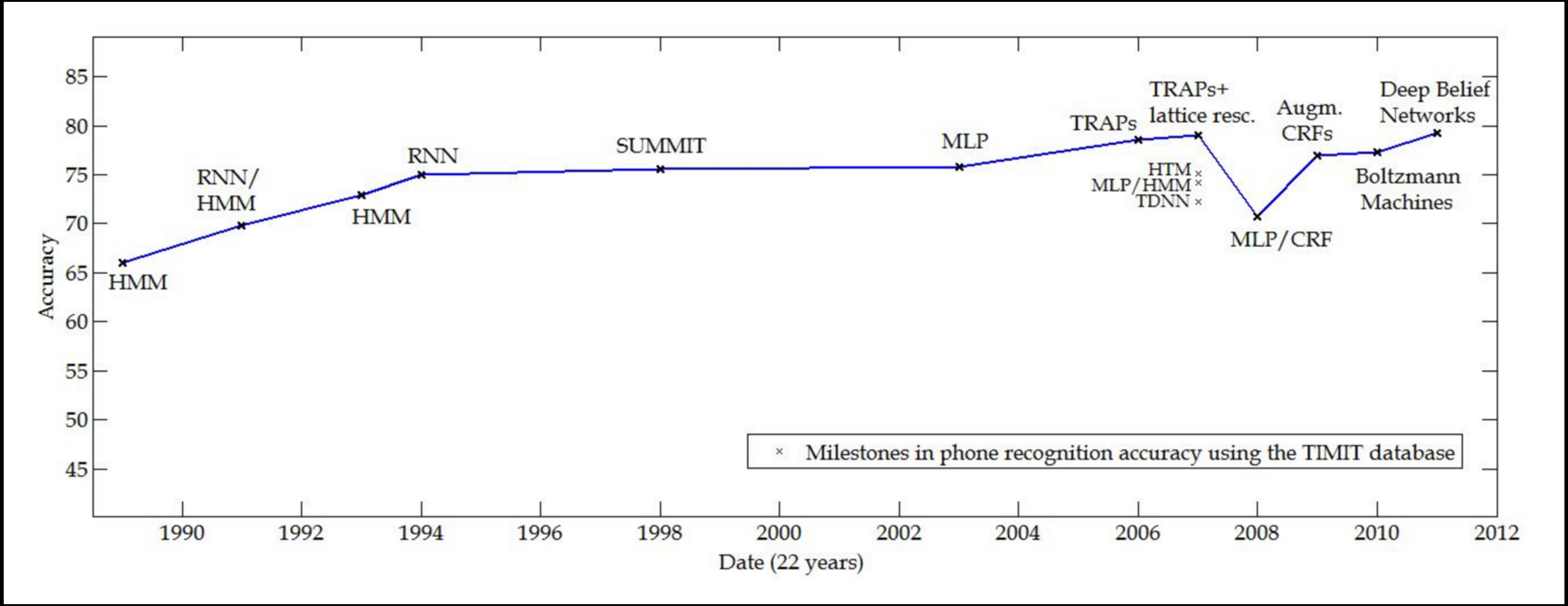
- To make good predictions, we sometimes need more context
- We need **long-term** memory capabilities without extending the network's recursion indefinitely (unscalable)



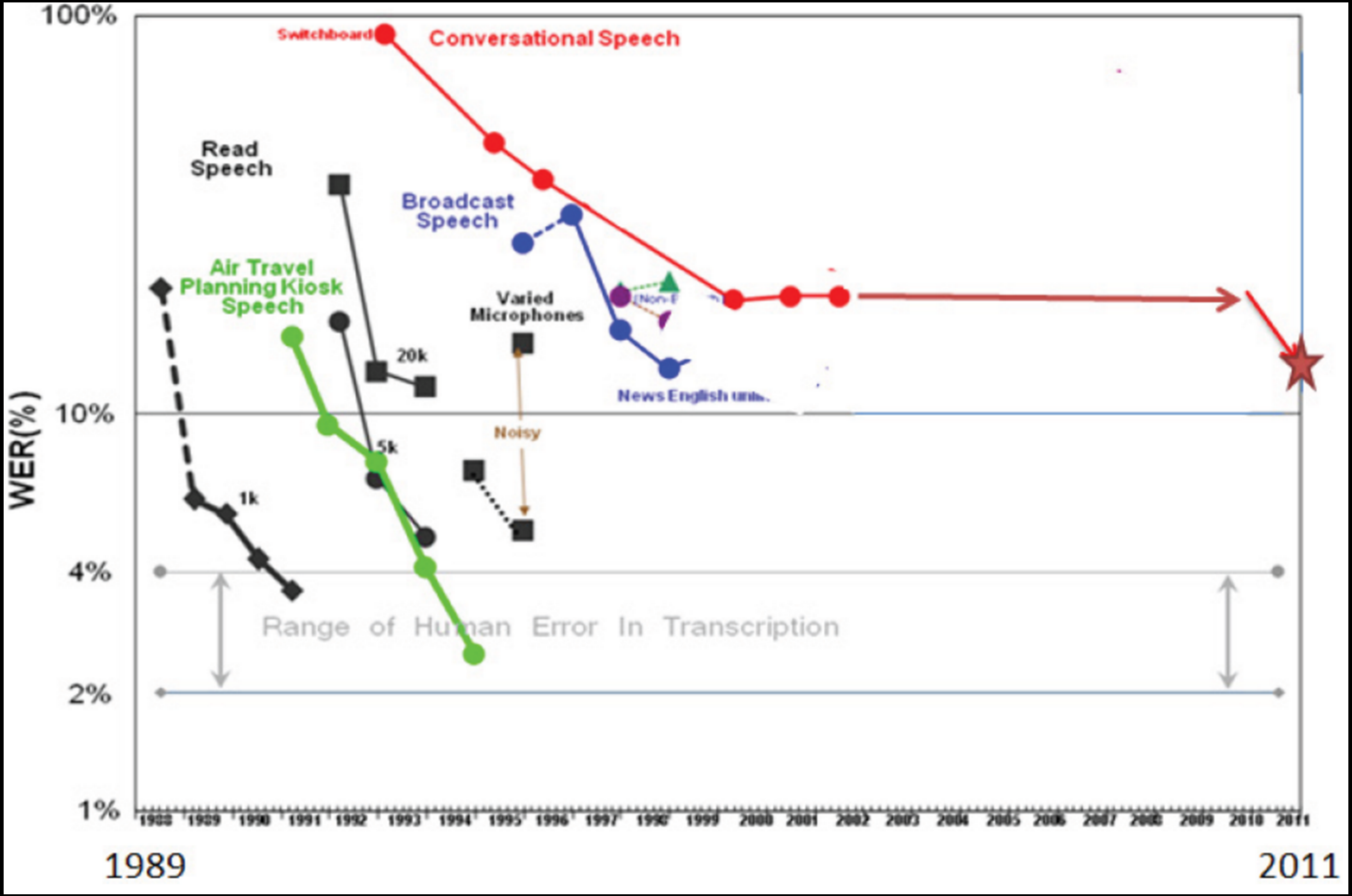
Disney, Finding Dory



Colah's Blog, "Understanding LSTM Networks"



Carla et. al., “Phone Recognition on the TIMIT Database”



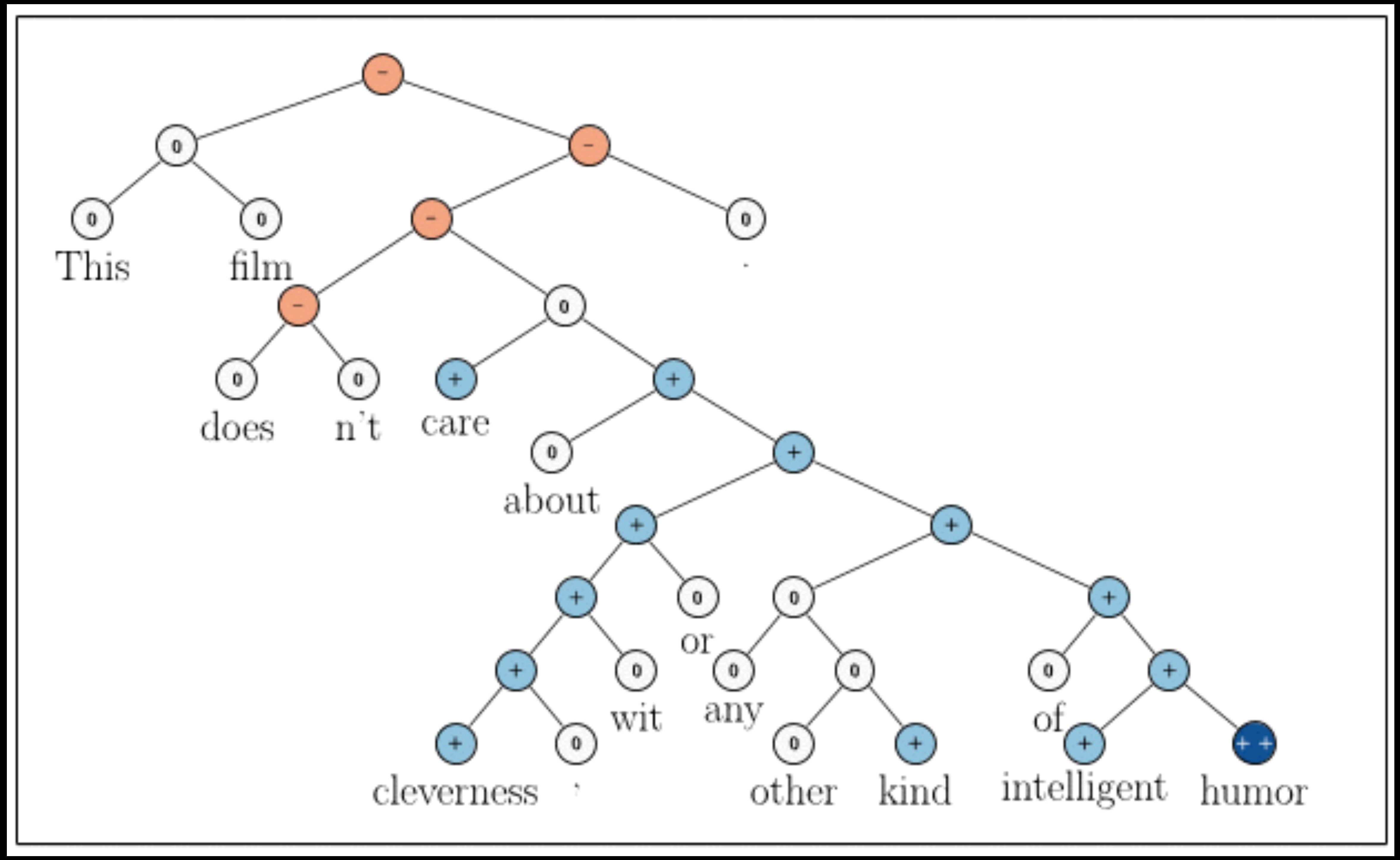
Deng et. al. “Deep Learning: Methods and Applications”

Beyond Multi Layer Perceptrons

Recursive Neural Tensor Network

(for sentiment analysis)

- Handles multiplicity in the data
    - Window/batch of events
    - i.e. a phrase of words
    - i.e. 24 frames of a video



Socher, 2013, “Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank"

Deep Neural Networks

Research in this field is still ***very*** active

So much is going on in this space now

Deep Neural Networks

State of the art optimizations that you can use out of the box

Dropout

Regularization technique:
randomly drop units & connections during training
dropout factor 0.5 found to perform well across many applications

Srivastava et. al., 2014

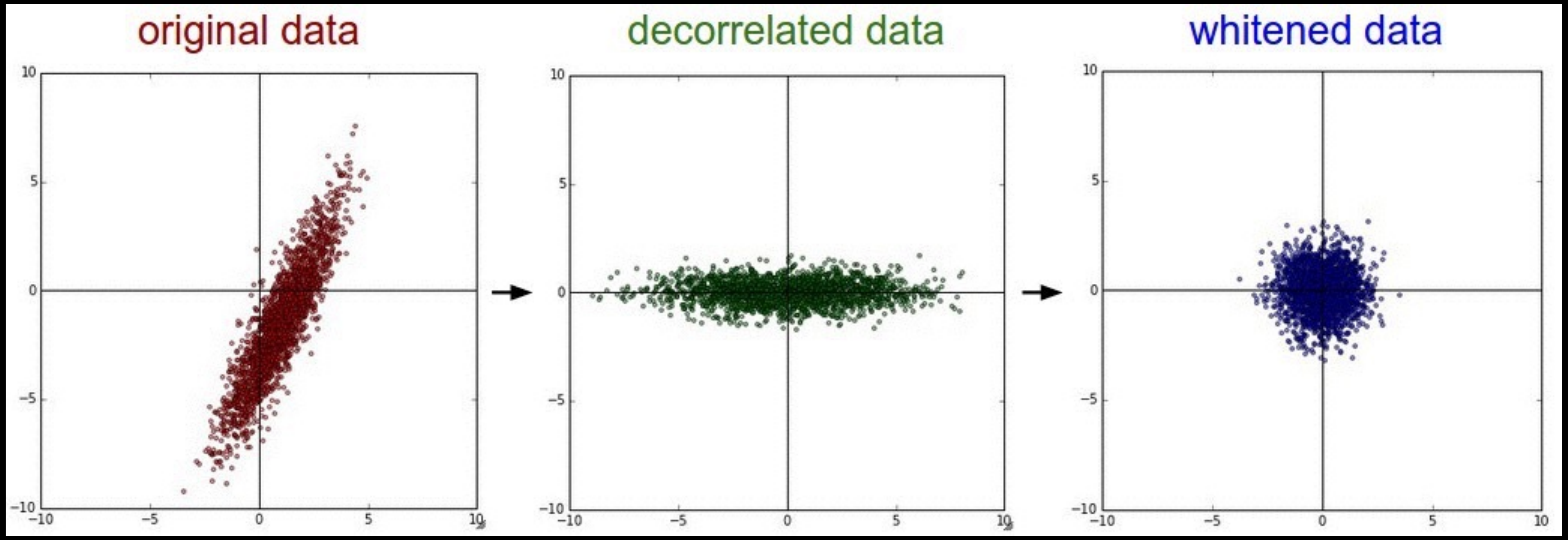
The Foothill Dragon Press

Deep Neural Networks

State of the art optimizations that you can use out of the box

PCA Whitening

Dimensionality Reduction
frequently used for noisy image inputs
changes the input vector into a white noise vector



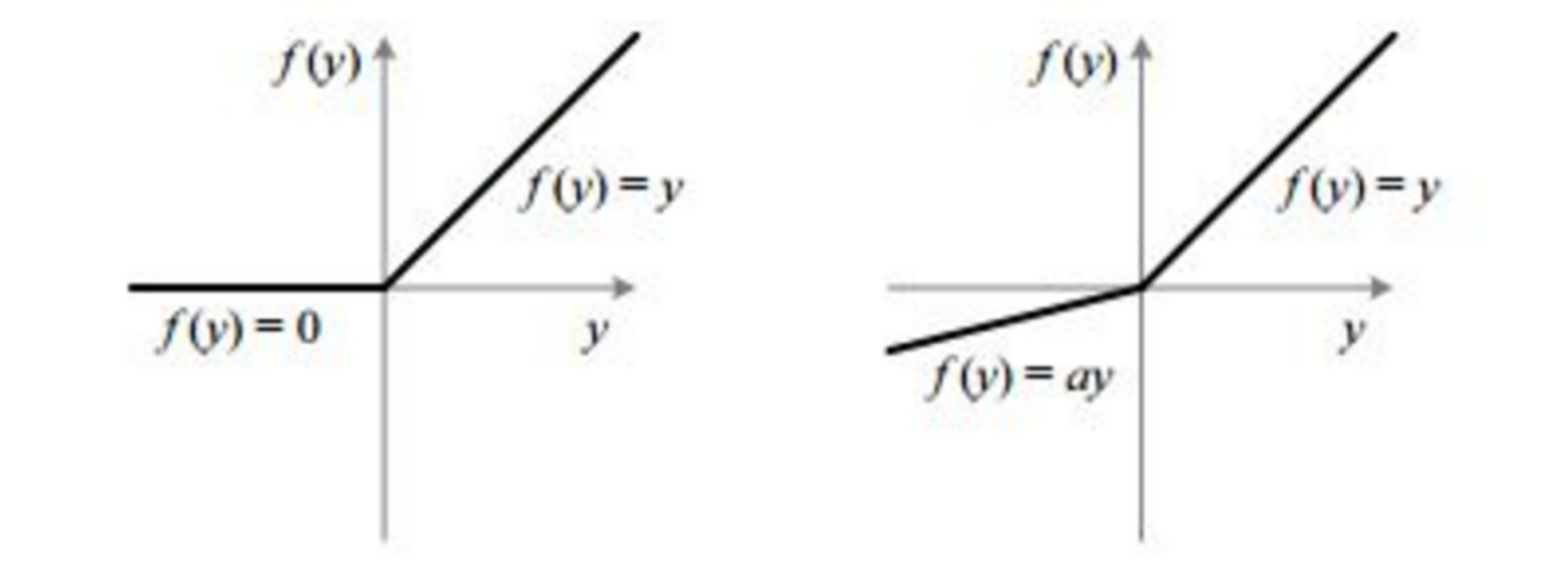
Stanford CS231n

Deep Neural Networks

State of the art optimizations that you can use out of the box

Leaky Rectified Linear Unit (ReLU) Activation Function

Popular activation function (other popular ones are sigmoid, tanh, ReLU)

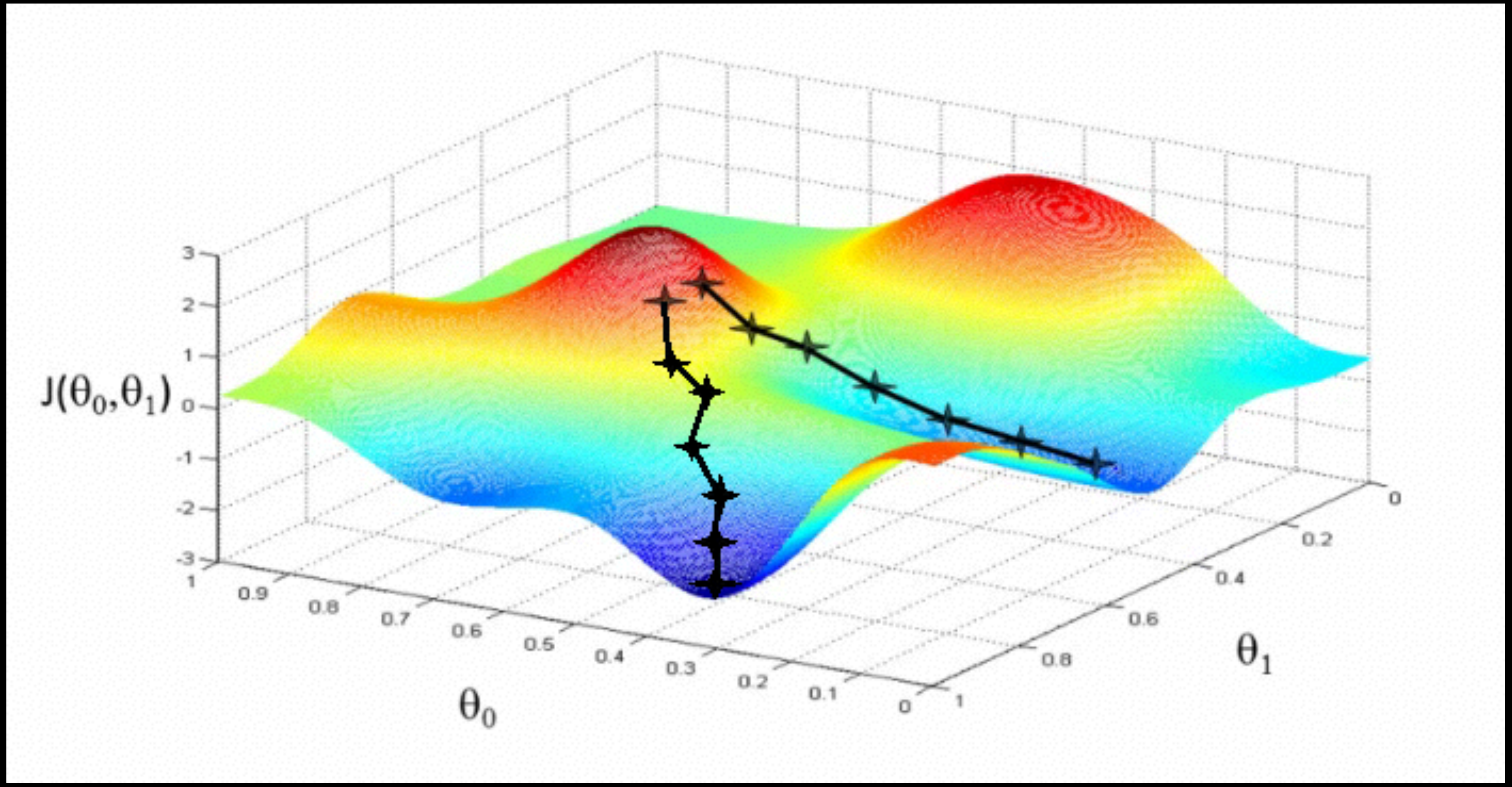


He et. al., 2015

Deep Neural Networks

State of the art optimizations that you can use out of the box

Loss function optimization methods



DatumBox, 2013

HOW TO PWN?

Attack Taxonomy

Applicable also to **online-learning** models that continuously learn from real-time test data

	Causative (Manipulative training samples)	**Exploratory** (Manipulative test samples)
Targeted	Training samples that move classifier decision boundary in an intentional direction	Adversarial input crafted to cause an intentional misclassification
Indiscriminate	Training samples that increase FP/FN → renders classifier unusable	n/a

DEMO

MNIST MISCLASSIFICATION

SENTIMENT MISCLASSIFICATION

Why can we do this?

vs.

BLINDSPOTS:

Statistical learning models don't **learn** concepts the same way that we do.

Adversarial Deep Learning

Intuitions

1. **Run input *x* through the classifier model (or substitute/approximate model)**
2. **Based on model prediction, derive a <u>perturbation tensor</u> that maximizes chances of misclassification:**
    1. Traverse the manifold to find blind spots in input space; or
    2. Linear perturbation in direction of neural network's cost function gradient; or
    3. Select only input dimensions with high saliency* to perturb by the model's Jacobian matrix
3. **Scale the perturbation tensor by some <u>magnitude</u>, resulting in the effective perturbation (δ_x) to *x***
    1. Larger perturbation == higher probability for misclassification
    2. Smaller perturbation == less likely for human detection

*** saliency:** amount of **influence** a selected dimension has on the entire model's output

Adversarial Deep Learning

Intuitions

Szegedy, 2013: Traverse the manifold to find blind spots in the input space

- Adversarial samples == pockets in the manifold
- Difficult to efficiently find by brute force (high dimensional input)
- Optimize this search, take gradient of input w.r.t. target output class

Adversarial Deep Learning

Intuitions

Goodfellow, 2015: Linear adversarial perturbation

- Developed a linear view of adversarial examples
- Can just take the cost function gradient w.r.t. the sample (x) and original predicted class (y)
- Easily found by backpropagation

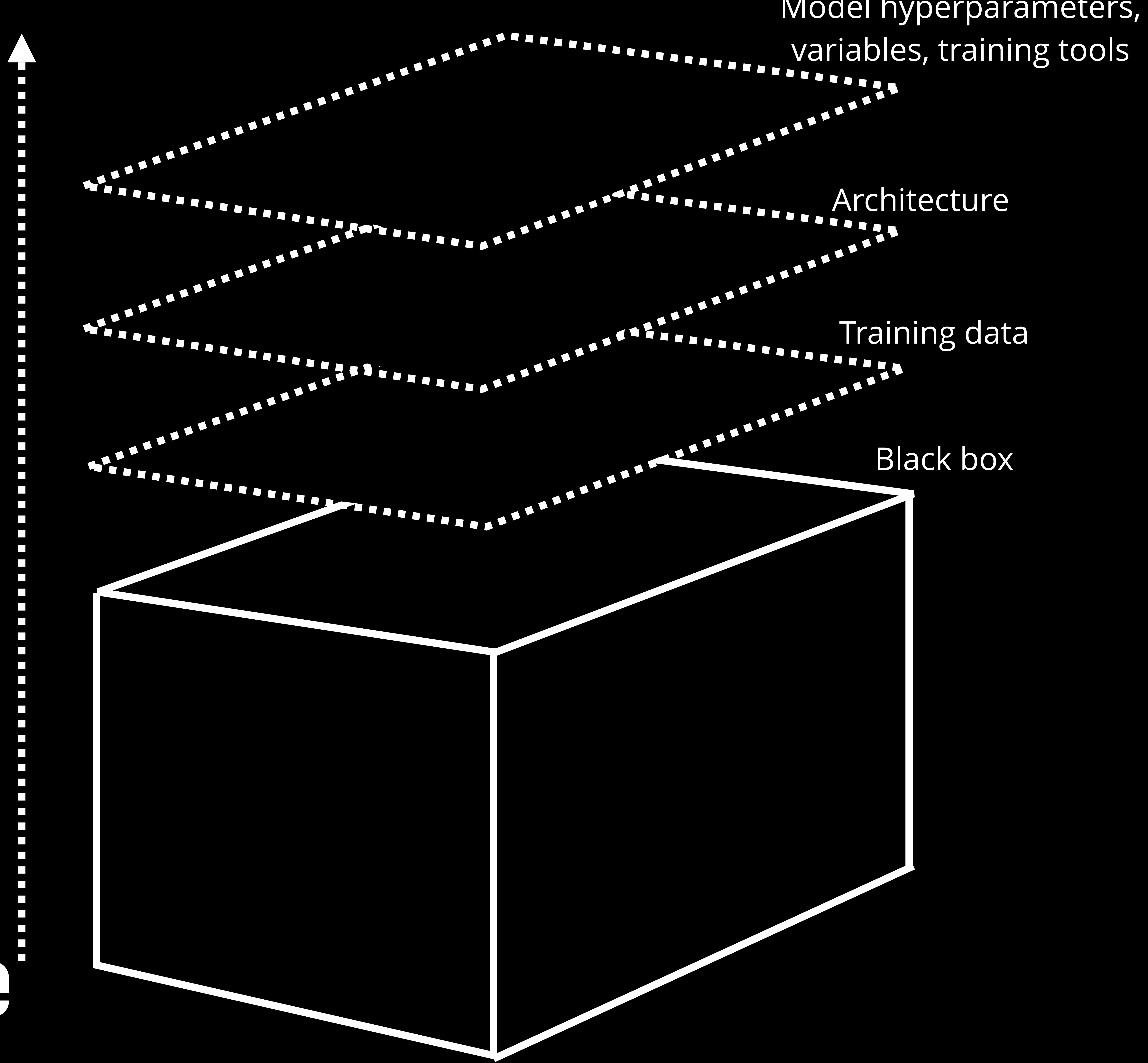
Adversarial Deep Learning

Intuitions

Papernot, 2015: Saliency map + Jacobian matrix perturbation

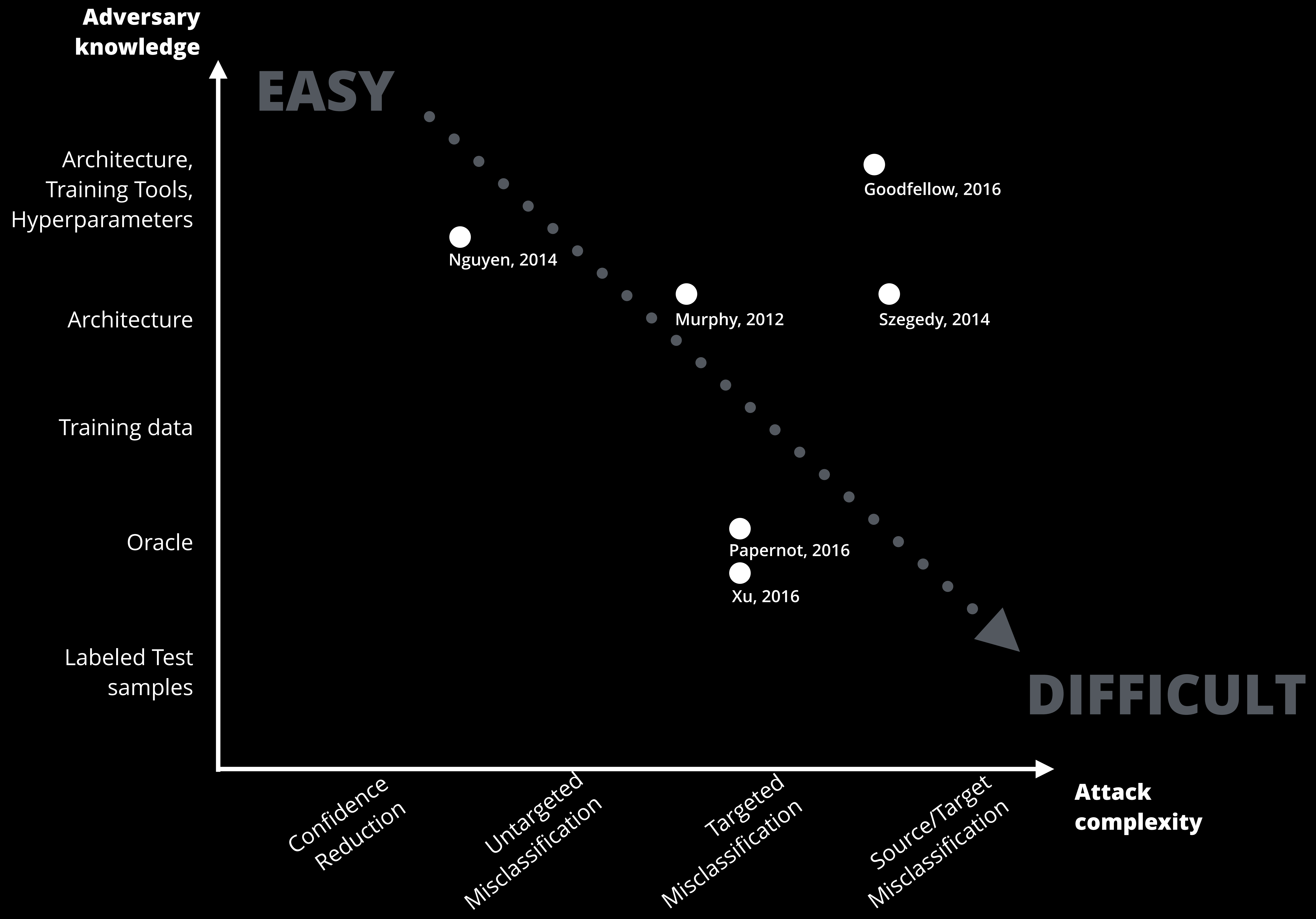
- More complex derivations for why the Jacobian of the learned neural network function is used
    - Obtained with respect to input features rather than network parameters
    - Forward propagation is used instead of backpropagation
- To reduce probability of human detection, only perturb the dimensions that have the greatest impact on the output (salient dimensions)

Increasing attacker knowledge



Threat Model: Adversarial Knowledge

Deep Neural Network Attacks



What can you do with limited knowledge?

- Quite a lot.
- Make good guesses: Infer the methodology from the task
  - Image classification: ConvNet
  - Speech recognition: LSTM-RNN
  - Amazon ML, ML-as-a-service etc.: Shallow feed-forward network
- What if you can’t guess?

STILL CAN PWN?

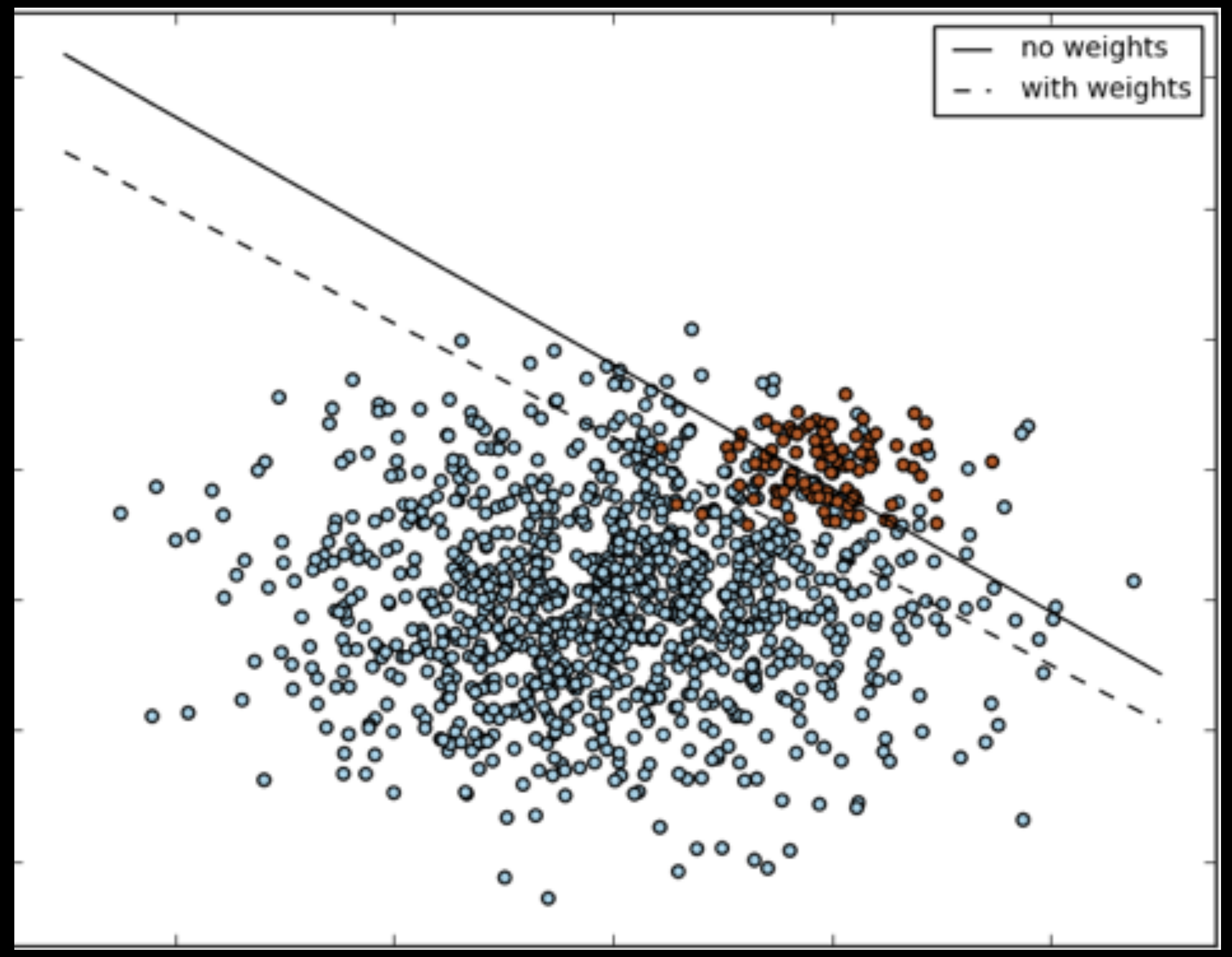
DEMO

Black box attack methodology

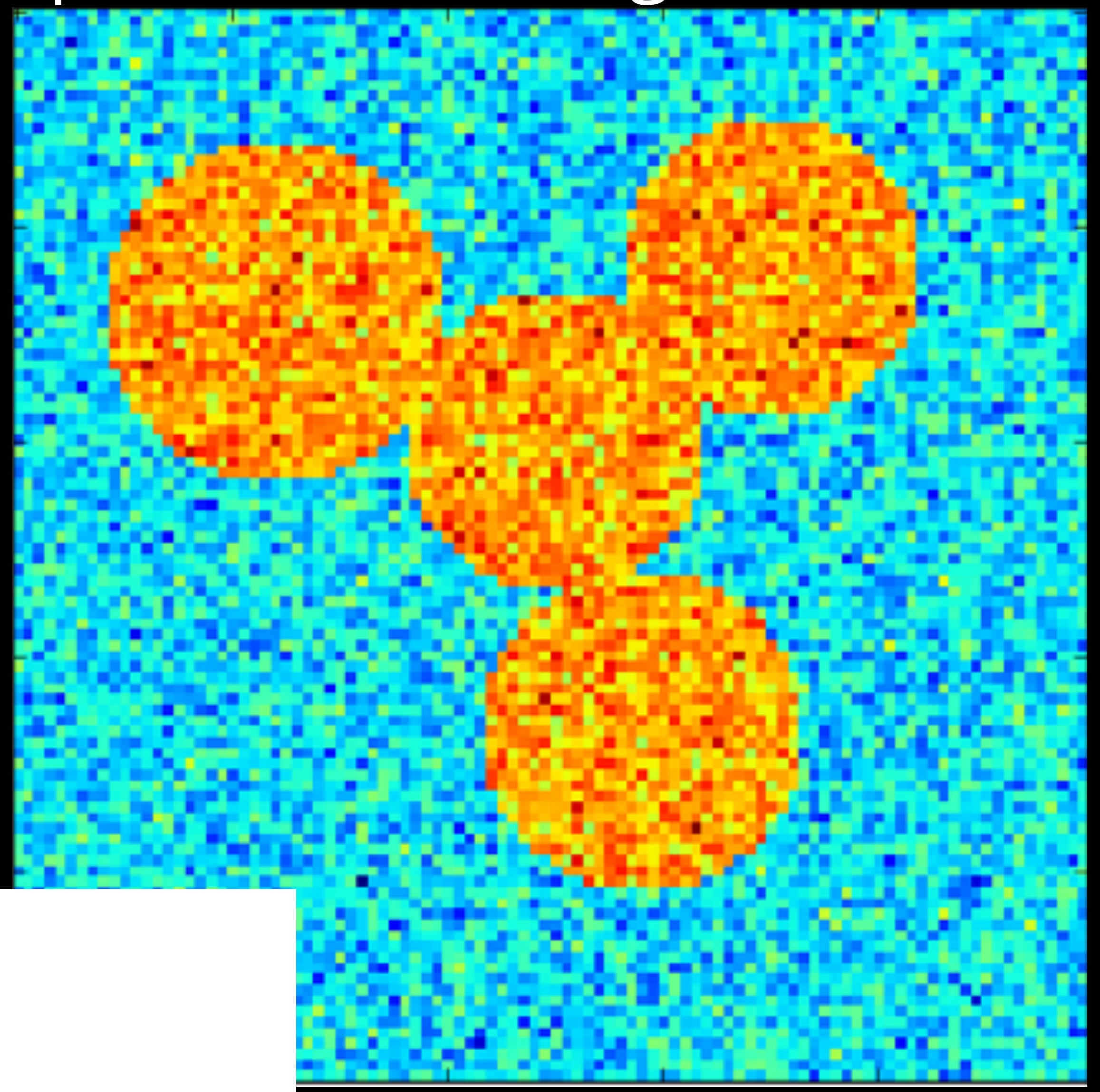
1. Transferability

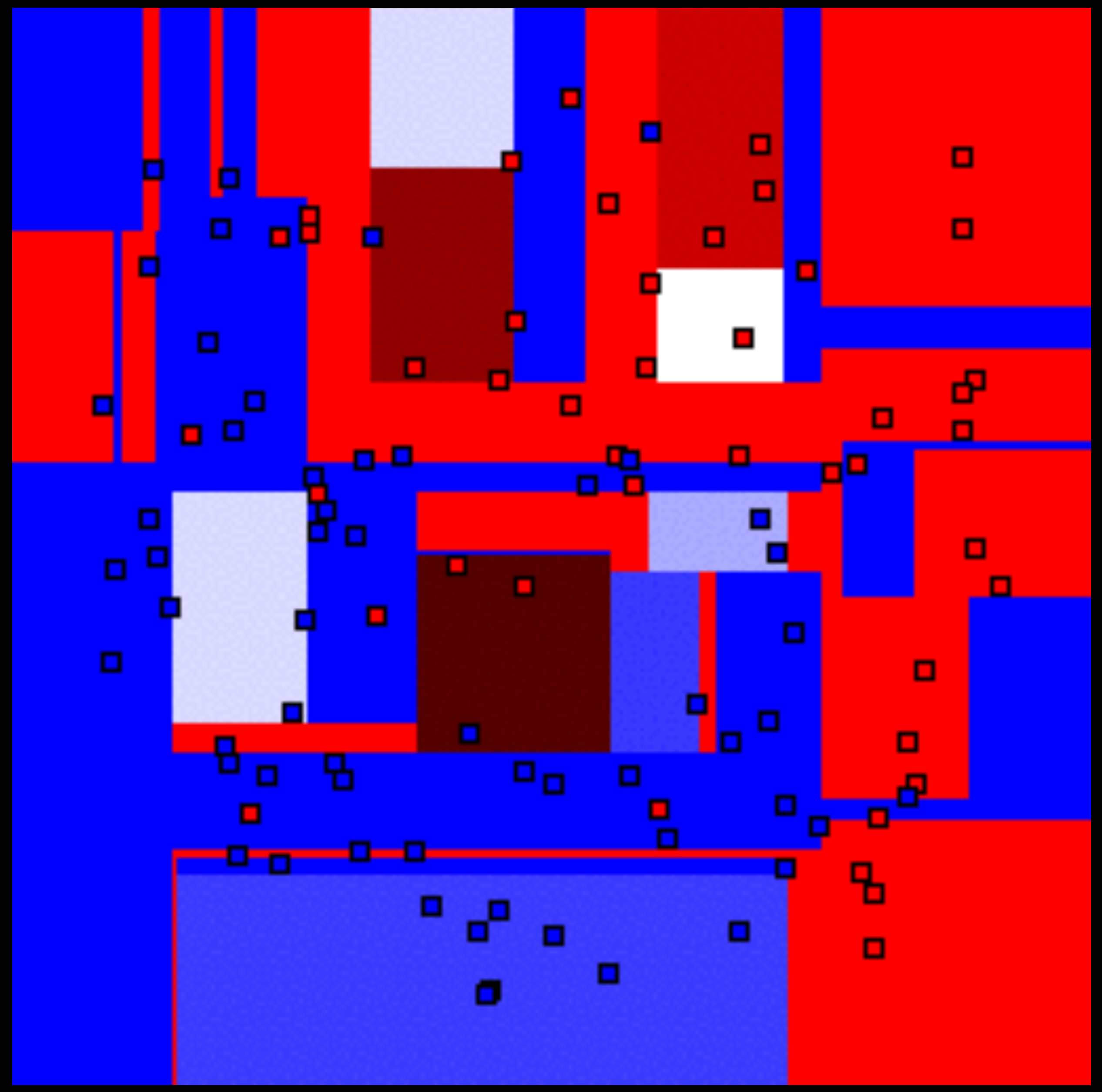
Adversarial samples that fool model A have a good chance of fooling a previously unseen model B

SVM, scikit-learn

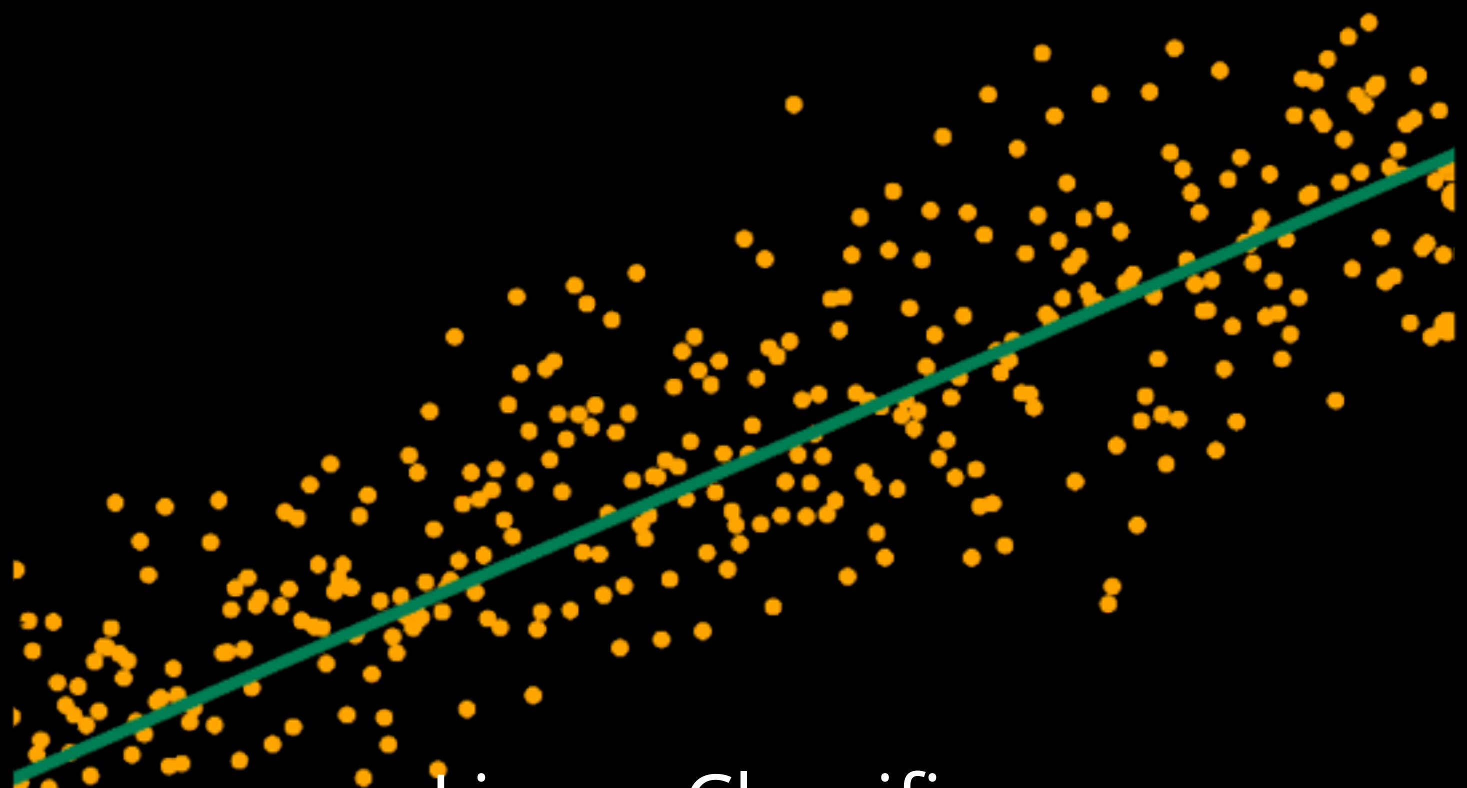


Spectral Clustering, scikit-learn

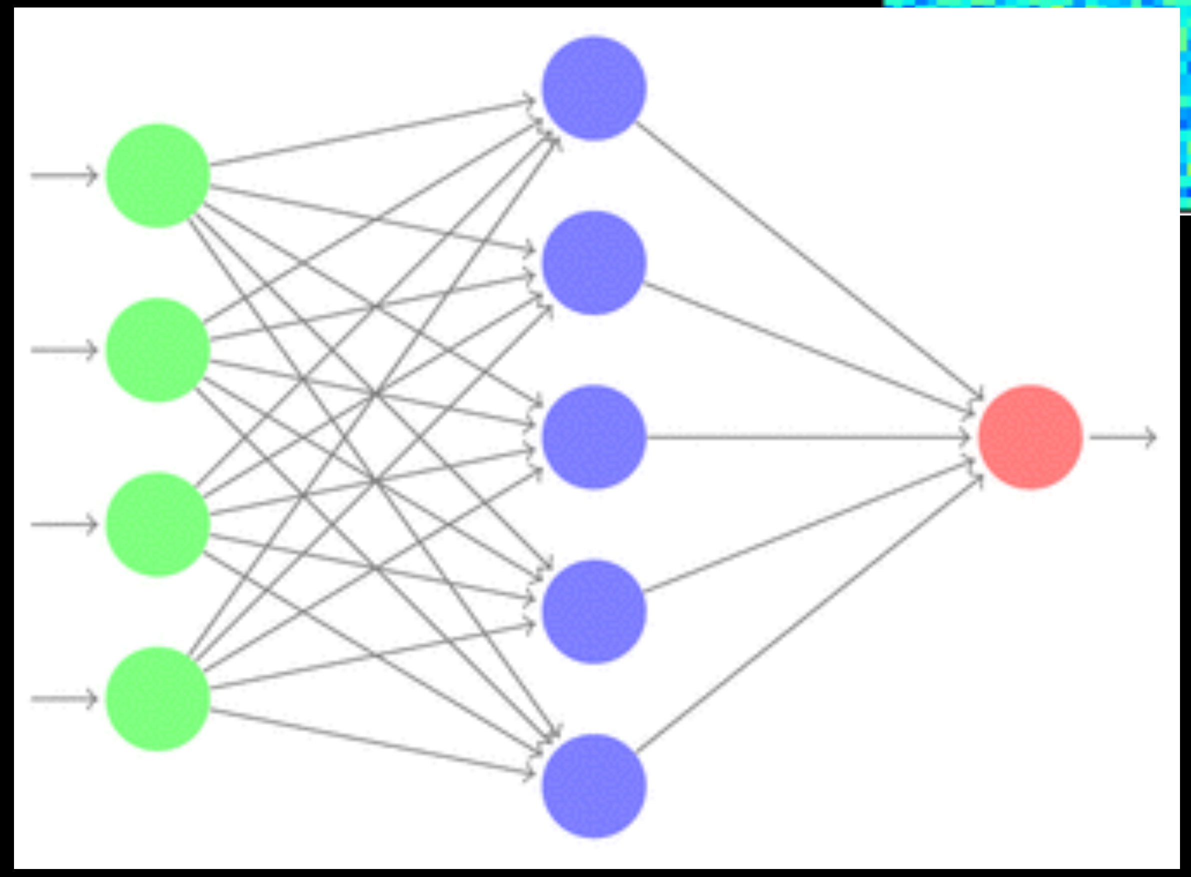




Decision Tree
Matt's Webcorner, Stanford



Linear Classifier
(Logistic Regression)

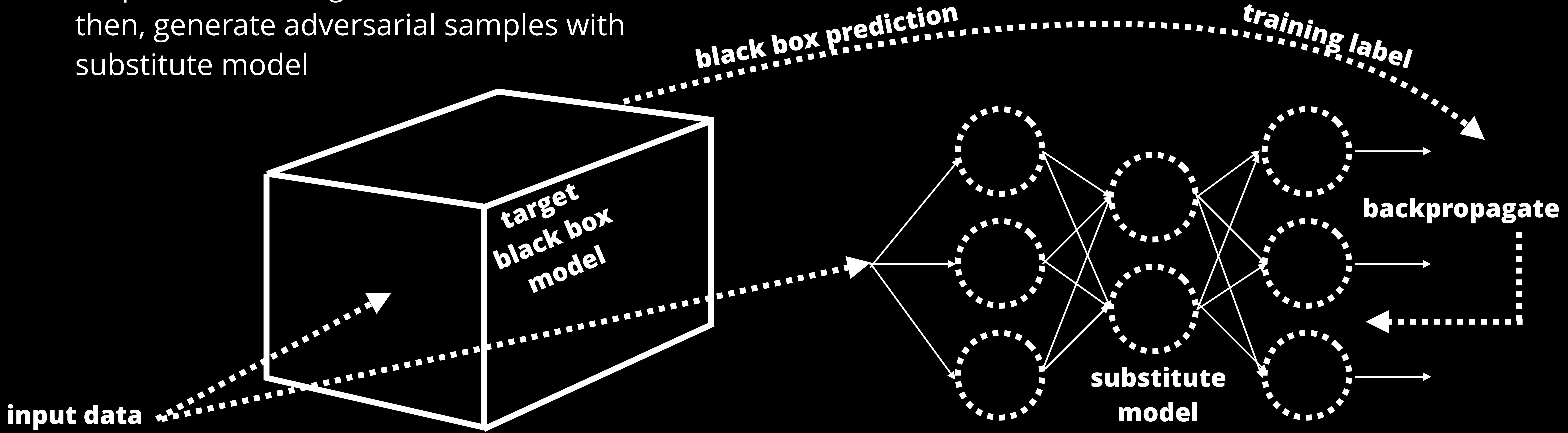


Feed Forward Neural Network

Black box attack methodology

2. Substitute model

train a new model by treating the target model's output as a training labels
then, generate adversarial samples with substitute model



Why is this possible?

- Transferability?
  - Still an open research problem
- Manifold learning problem
  - Blind spots
  - Model vs. Reality dimensionality mismatch
- **IN GENERAL:**
  - Is the model not learning anything at all?

What this means for us

- Deep learning algorithms (Machine Learning in general) are susceptible to manipulative attacks
    - Use with caution in critical deployments
- Don’t make false assumptions about what/how the model learns
- Evaluate a model’s **adversarial resilience** - not just accuracy/precision/recall
- Spend effort to make models more robust to tampering

Defending the machines

- Distillation
  - Train model 2x, feed first DNN output logits into second DNN input layer
- Train model with adversarial samples
  - i.e. ironing out imperfect knowledge learnt in the model
- Other miscellaneous tweaks
  - Special regularization/loss-function methods (simulating adversarial content during training)
  - *DATAGRAD*

DEEP-PWNING

“metasploit for machine learning”

WHY DEEP-PWNING?

- **lol why not**
- “Penetration testing” of statistical/machine learning systems
- Train models with adversarial samples for increased robustness

DEMO

PLEASE PLAY WITH IT &

CONTRIBUTE!

Deep Learning and Privacy

- **Deep learning also sees challenges in other areas relating to security & privacy**
- **Adversary can reconstruct training samples from a trained black box DNN model** (Fredrikson, 2015)
- **Can we precisely control the learning objective of a DNN model?**
- **Can we train a DNN model without the training agent having complete access to all training data?** (Shokri, 2015)

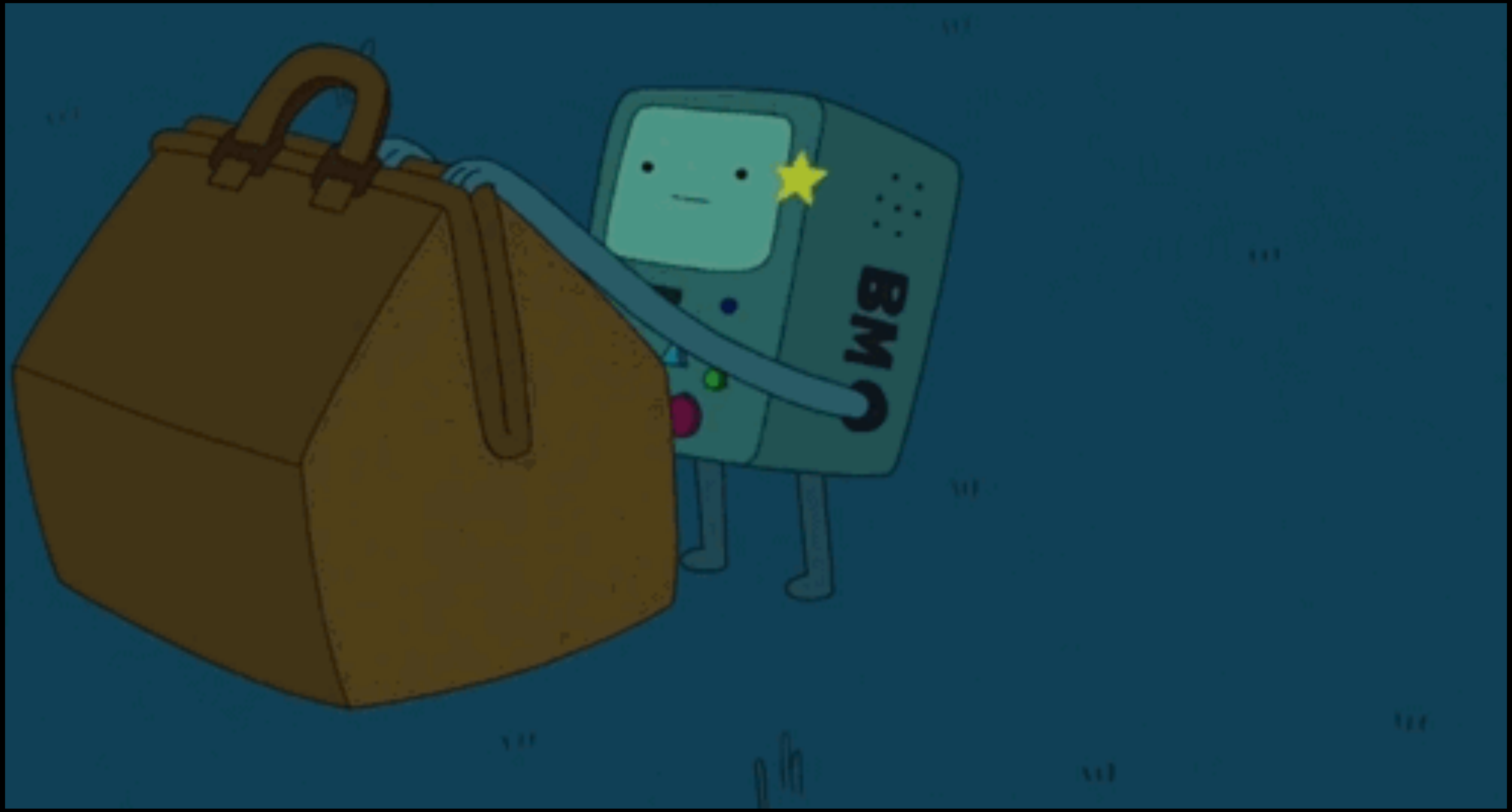
WHY IS THIS IMPORTANT?

WHY DEEP-PWNING?

- **MORE CRITICAL SYSTEMS RELY ON MACHINE LEARNING → MORE IMPORTANCE ON ENSURING THEIR ROBUSTNESS**

- **WE NEED PEOPLE WITH BOTH SECURITY AND STATISTICAL SKILL SETS TO DEVELOP ROBUST SYSTEMS AND EVALUATE NEW INFRASTRUCTURE**

LEARN IT OR BECOME IRRELEVANT



@cchio

MLHACKER