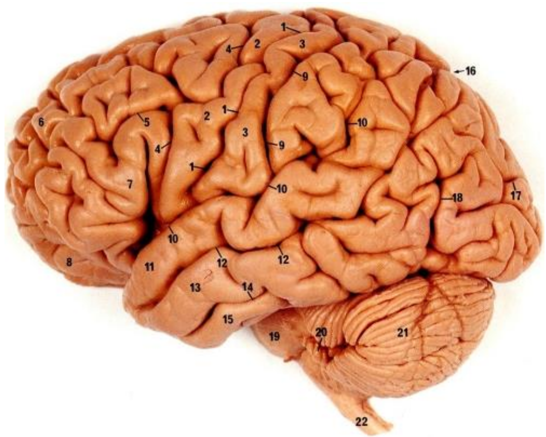
CSCI 561 - Foundation for Artificial Intelligence

Discussion Section (Week 12) NN and Deep Learning

PROF WEI-MIN SHEN SHEN@ISI.EDU

Real Neural Networks in Brain



Human brain is divided into regions with some functional specialization

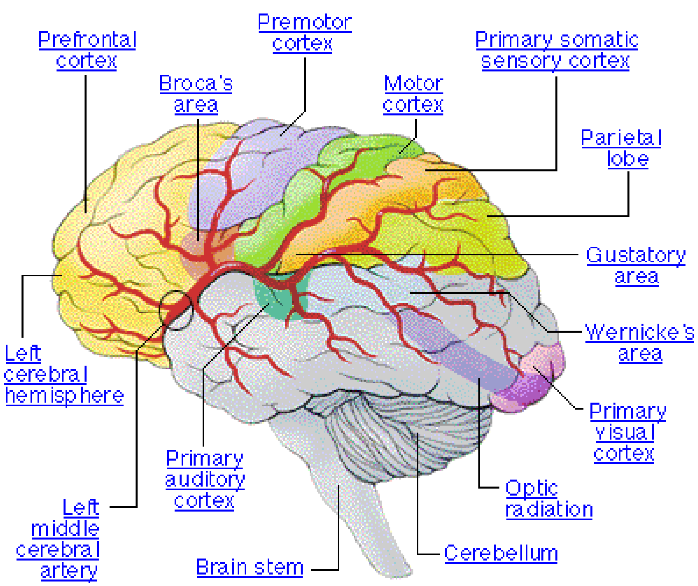
- E.g., Wernicke's and Broca's areas for language

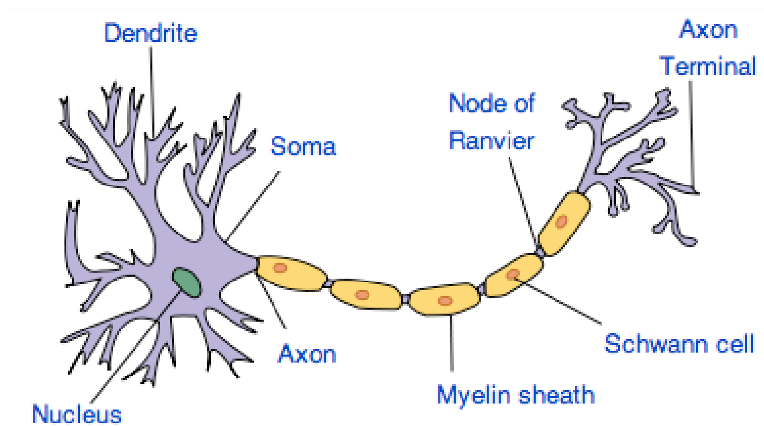
Computation is driven by very large networks of rather slow *neurons* connected via *synapses*

- 10^{11} neurons of > 20 types
- 10^{14} synapses
- 1-10ms cycle time
- Signals are noisy spike trains of electrical potential

In AI/ML, models of neurons are highly simplified

New Discovery: Glial Cells in Brain boost learning

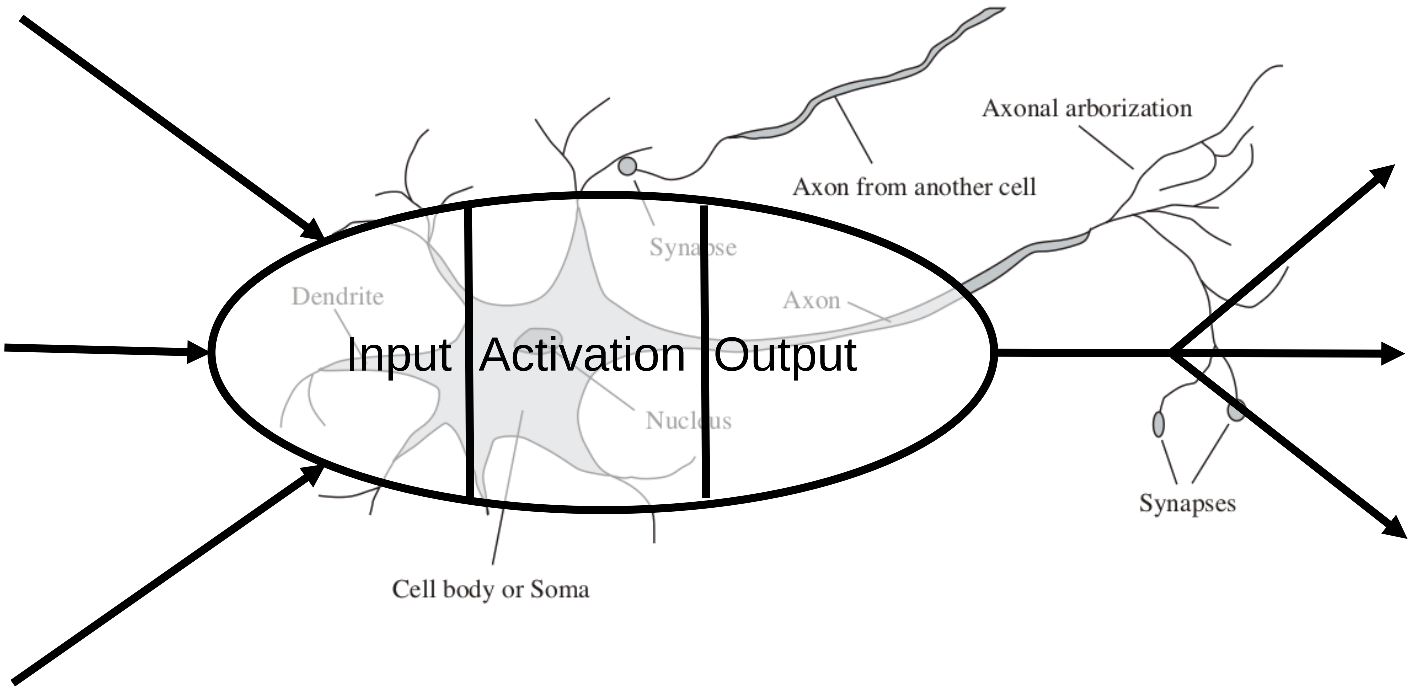




Artificial Neurons

Connectionism

- Activation is passed along links from neuron to neuron



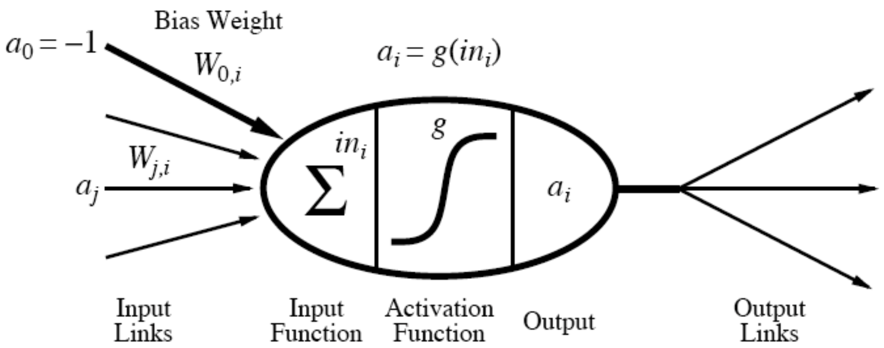
Neuron becomes a computational unit

- Output is determined as a function of incoming activation

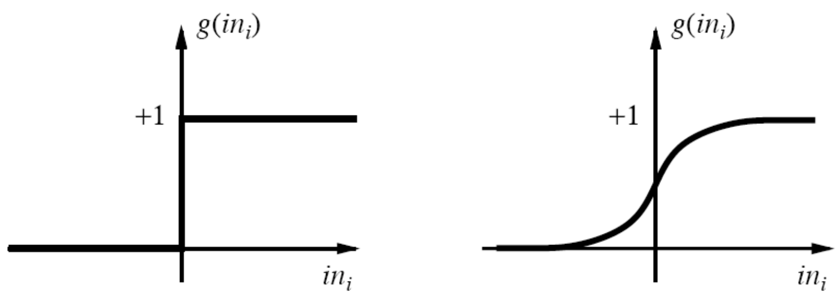
A modern view of "Units"

Output is a "squashed" linear function of inputs:

$a_i \leftarrow g(in_i) = g\left(\Sigma_j W_{j,i} a_j\right)$



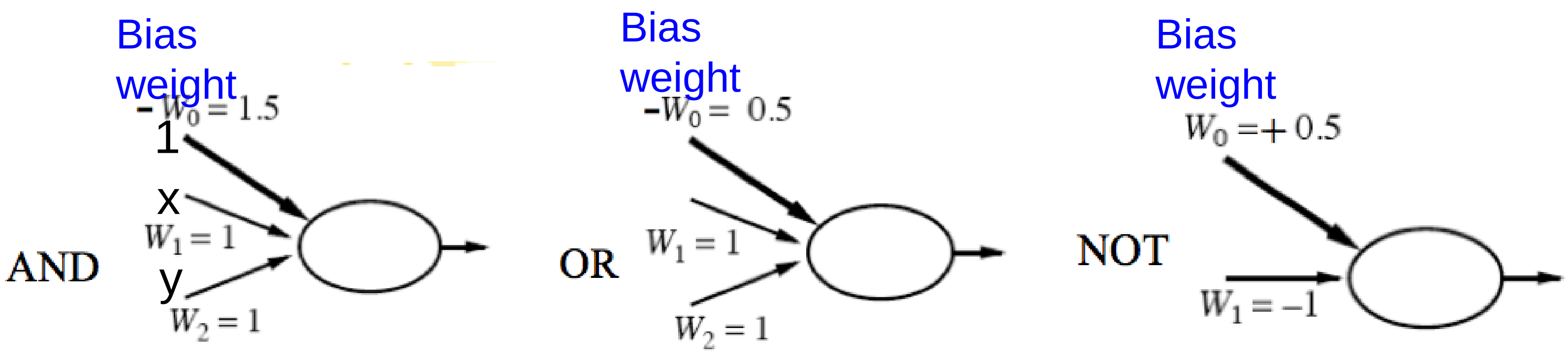
Activation function can be *step/threshold* or *sigmoid*



Bias weight sets threshold for *linear threshold unit*

Implementing Logical Functions

Any Boolean function can be implemented in a network of linear threshold units (*perceptrons*)



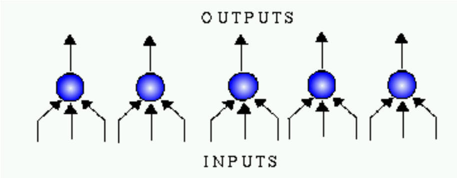
But *single layer networks* are limited

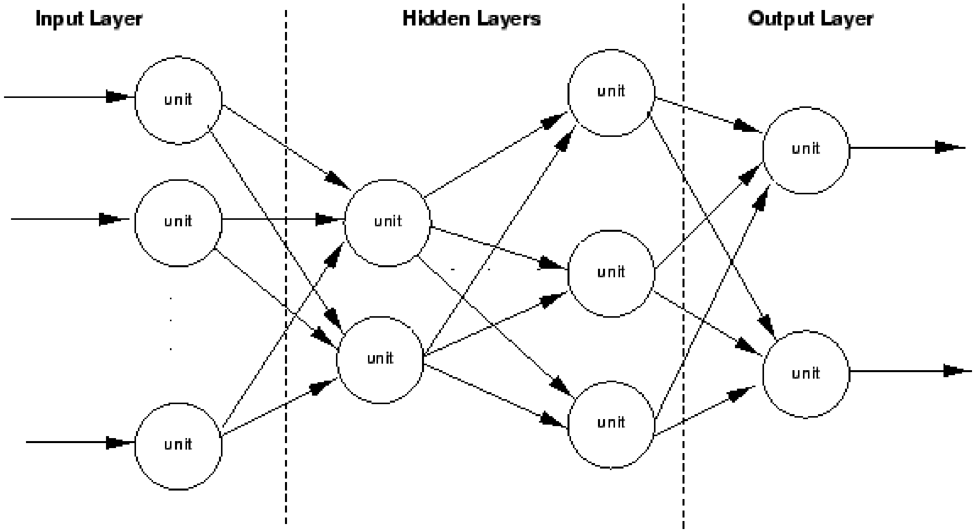
- E.g., cannot compute XOR
- Minsky & Papert essentially killed off neural network field for a decade by showing this in 1969

Variety of Network Structures

Feed-forward networks

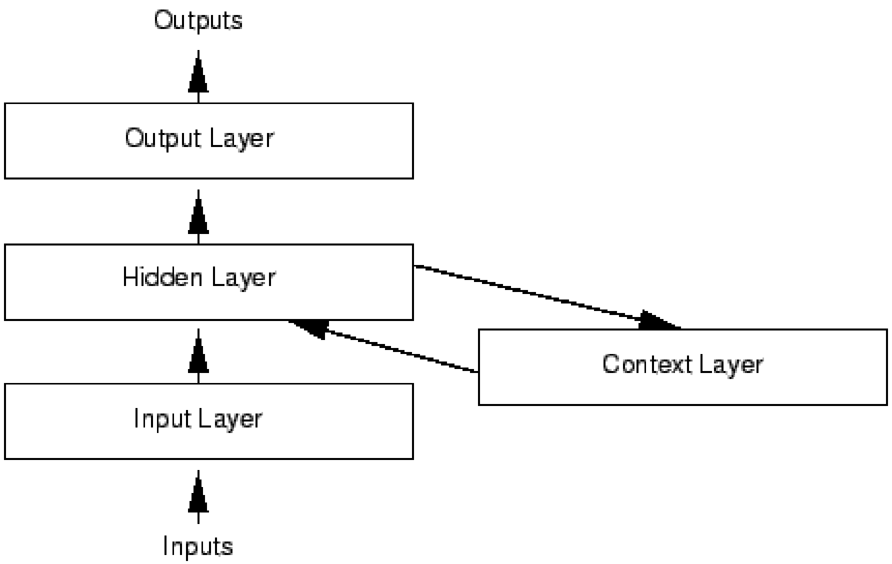
- May be single- or multi-layer
- Implement functions/reflexes
  - No internal state





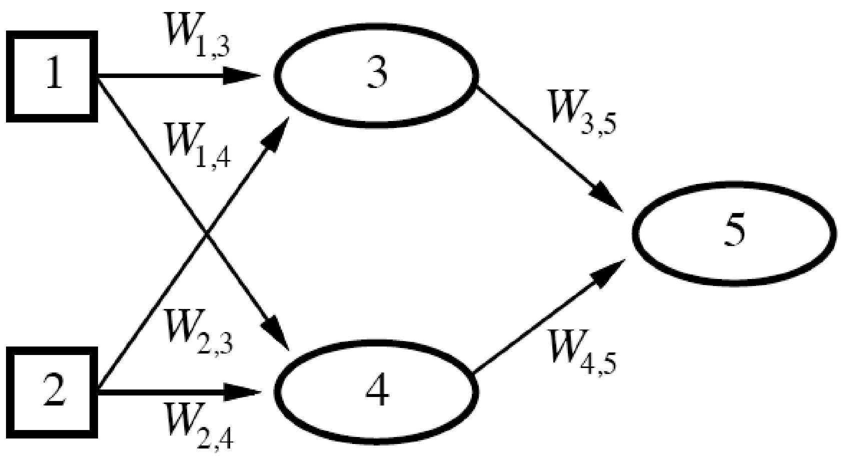
Recurrent networks

- Directed cycles with delays
  - Internal state, like flip-flops
  - Dynamical systems
- Hopfield networks
  - LTUs w/ symmetric weights
  - Minimize overall energy
- Boltzmann machines
  - Stochastic, simulated annealing



Feed-Forward Example

Feed-forward networks provide a parameterized family of nonlinear functions



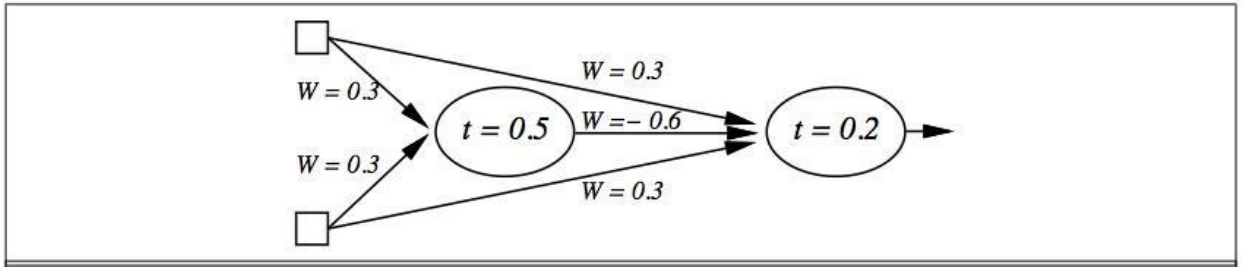
$a_5 = g(W_{3,5}a_3 + W_{4,5}a_4)$

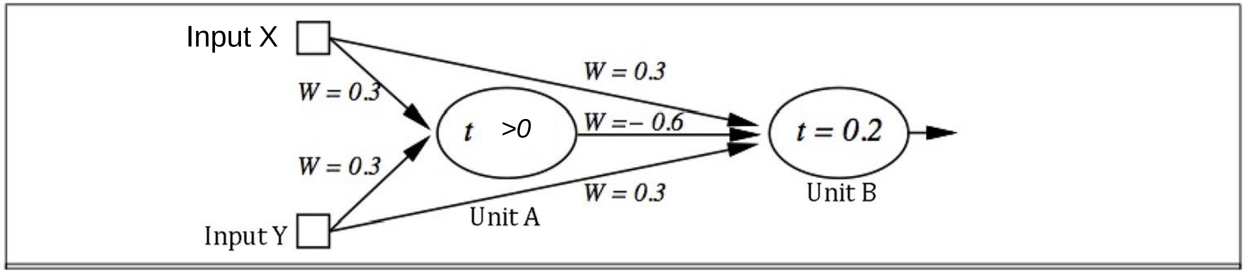
$= g(W_{3,5}g(W_{1,3}a_1+W_{2,3}a_2) + W_{4,5}g(W_{1,4}a_1+W_{2,4}a_2))$

Learning occurs by adjusting weights

Neural Network (exercise)

Draw the neural network to solve the XOR function for two inputs. Specify what type of unit you are using.





When input X = 0 and input Y = 0, what does the Unit A output? What does the Unit B output?

When input X = 0 and input Y = 1, what does the Unit A output? What does the Unit B output?

When input X = 1 and input Y = 0, what does the Unit A output? What does the Unit B output?

When input X = 1 and input Y = 1, what does the Unit A output? What does the Unit B output?

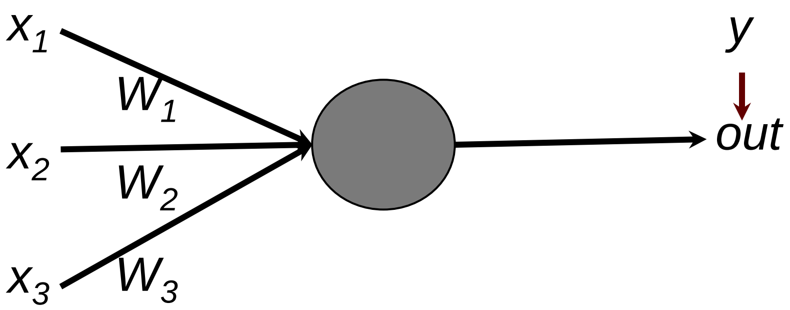
What Boolean function does this Neural Network compute?

Perceptron Learning

Learn by adjusting weights on each iteration to reduce error

- Gradient descent on squared error $(y - out)^2$
- Update rule for weights of a threshold function:
  $W_j' = W_j + \alpha\,(y - out)\,x_j$
- Comparable rules exist for differentiable activation functions

This *perceptron learning rule* is guaranteed to converge to a consistent function if the data is linearly separable

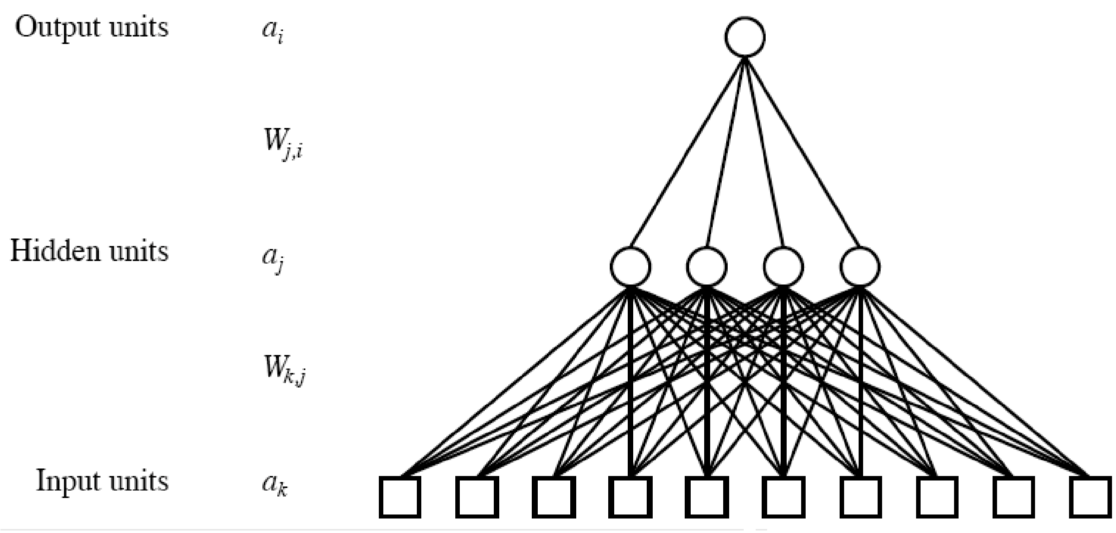


Multilayer Neural Networks

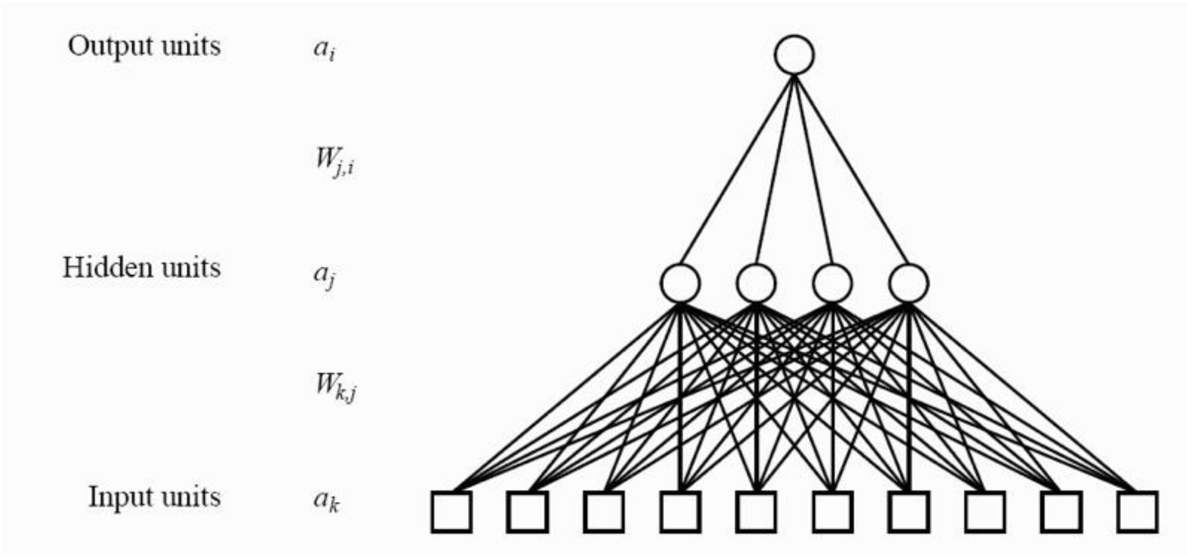
Units typically fully connected across layers unless have domain-specific knowledge to guide

Number of hidden units either chosen by hand or found via search

- If too many, network will memorize input (overfitting)
- If too few, function will be difficult to represent
- *As with Goldilocks, need it just right (to generalize)*



Multilayer network Learning (Back-Propagation)



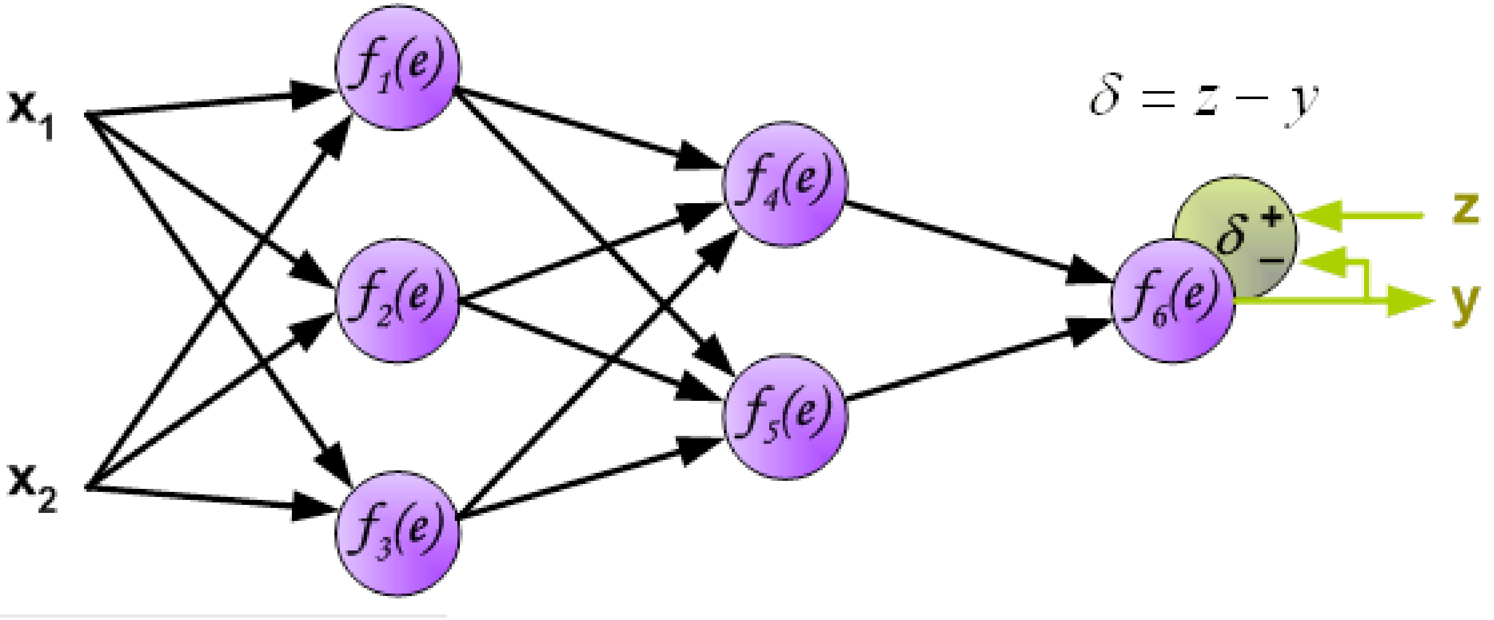
$$W_{ji} = W_{ji} + \alpha * a_j * \delta_i$$

$$\delta_i = (T_i - O_i) * g'(in_i)$$

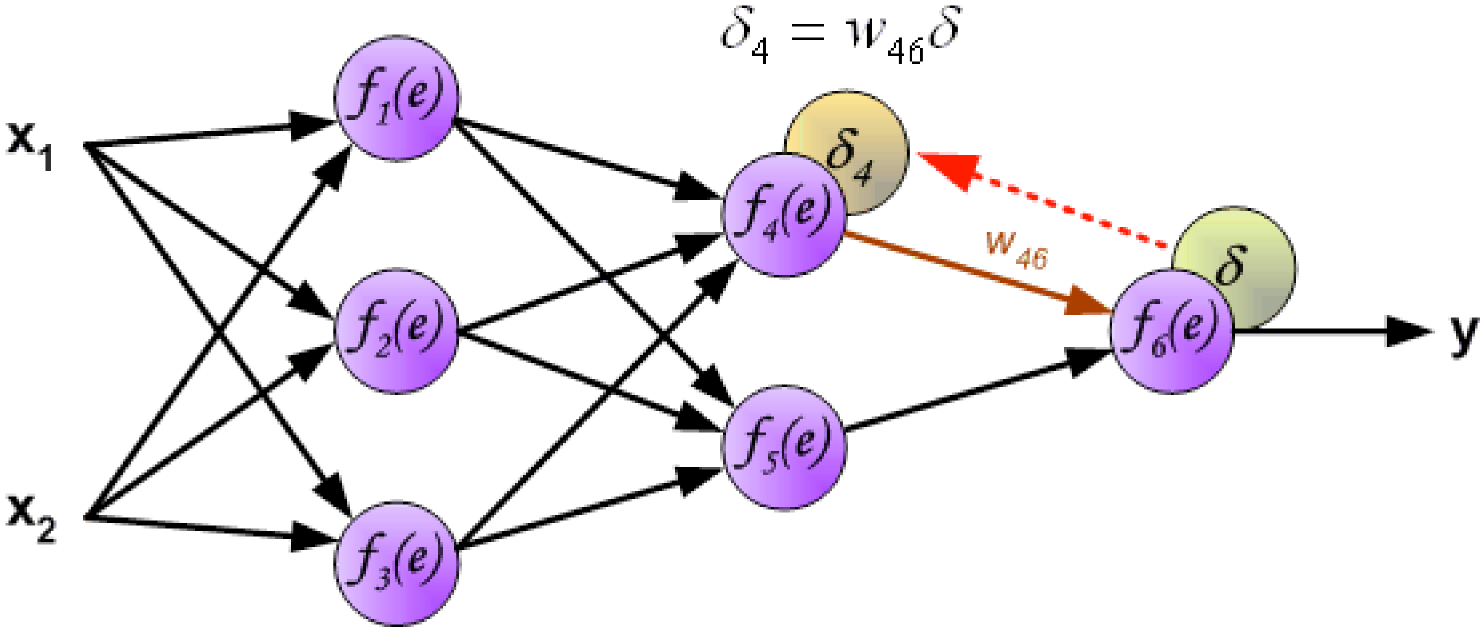
$$W_{kj} = W_{kj} + \alpha * a_k * \delta_j$$

$$\delta_j = g'(in_j) \sum W_{ji} \delta_i$$

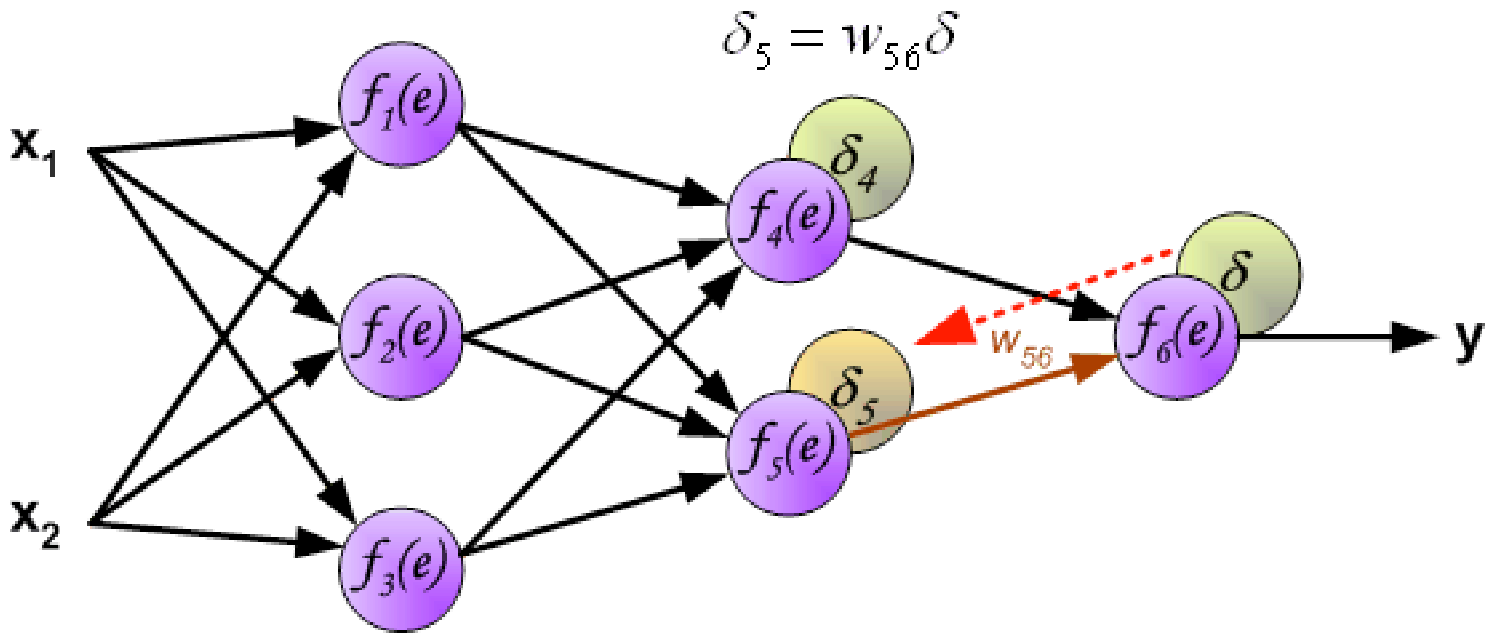
Example of Back-propagation



Example of Back-propagation

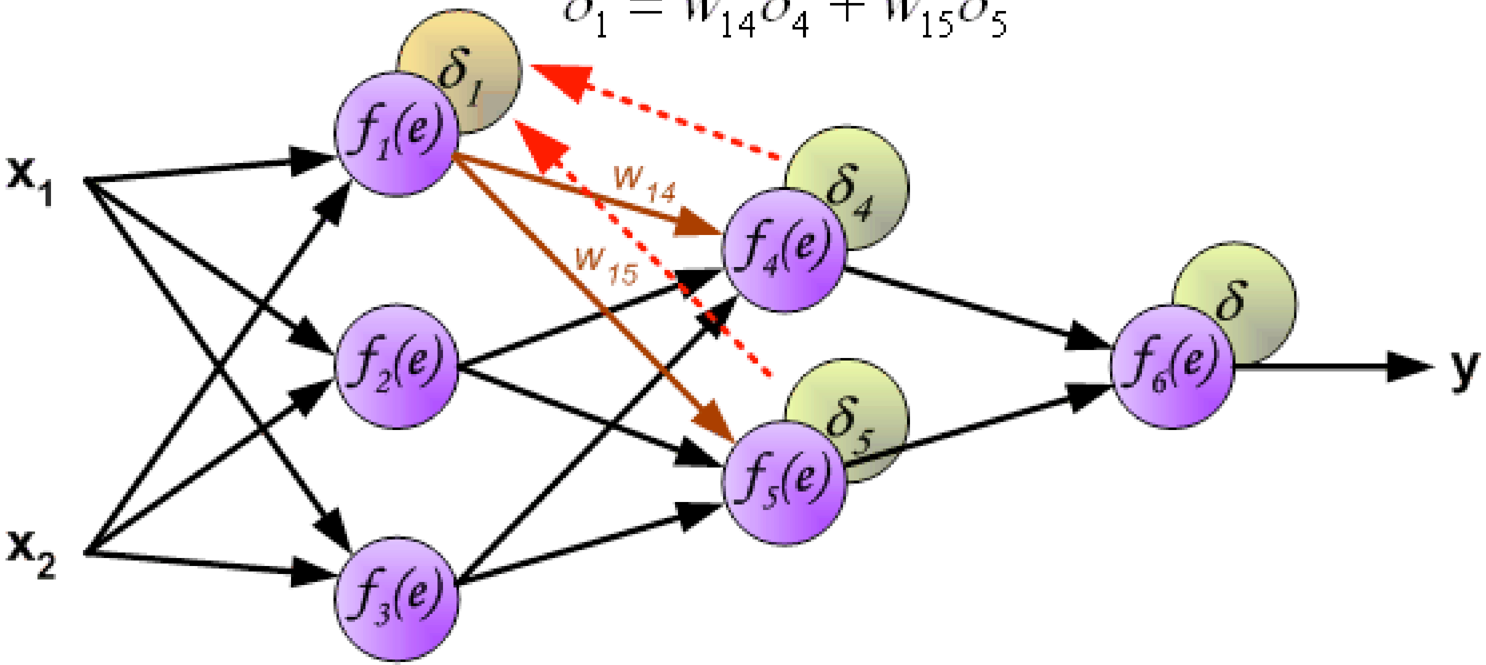


Example of Back-propagation



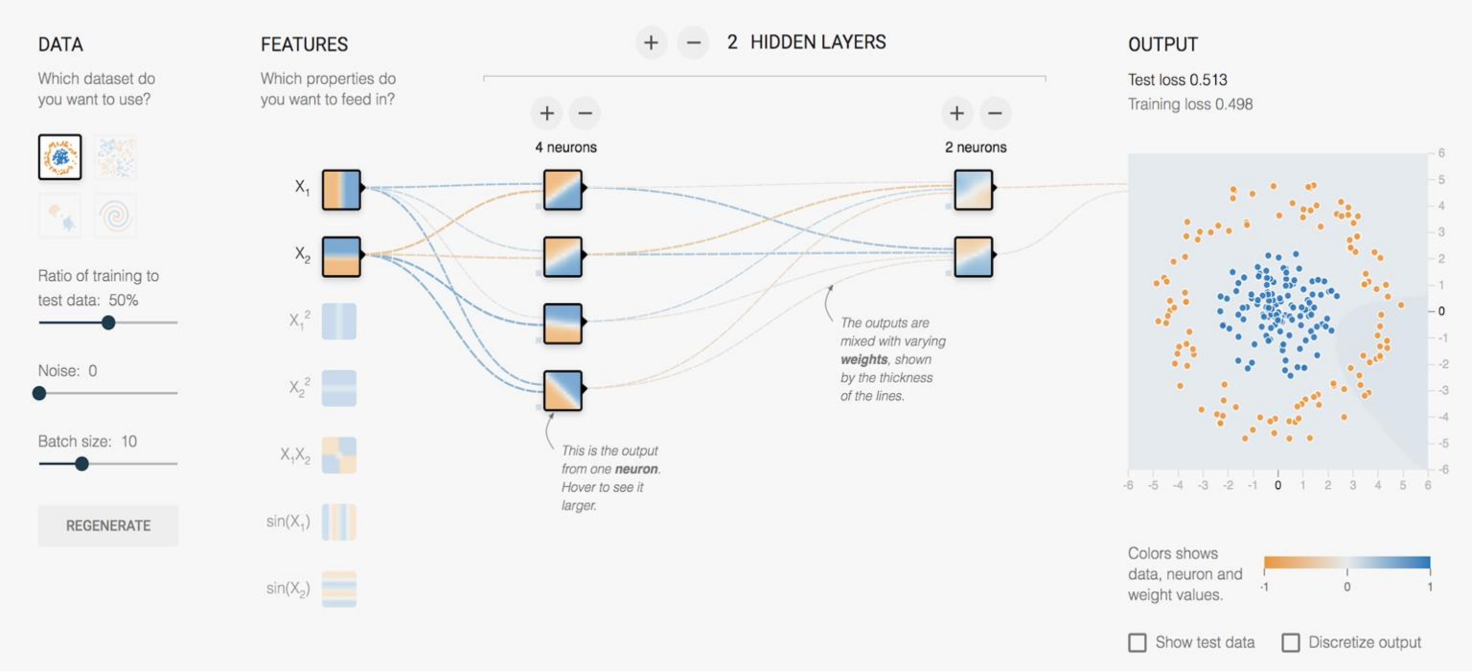
Example of Back-propagation

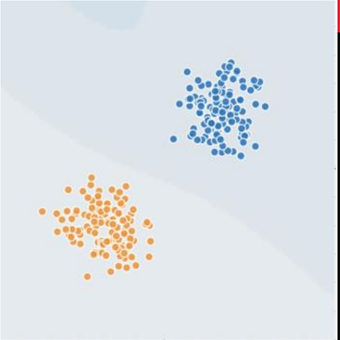
$$\delta_1 = w_{14}\delta_4 + w_{15}\delta_5$$

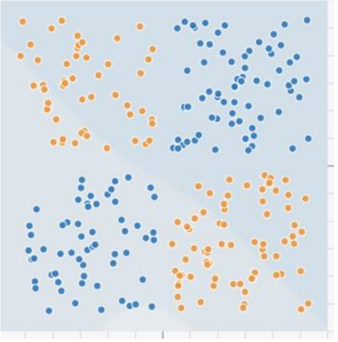


Google's Neural Network Playground

https://www.playground.tensorflow.org/

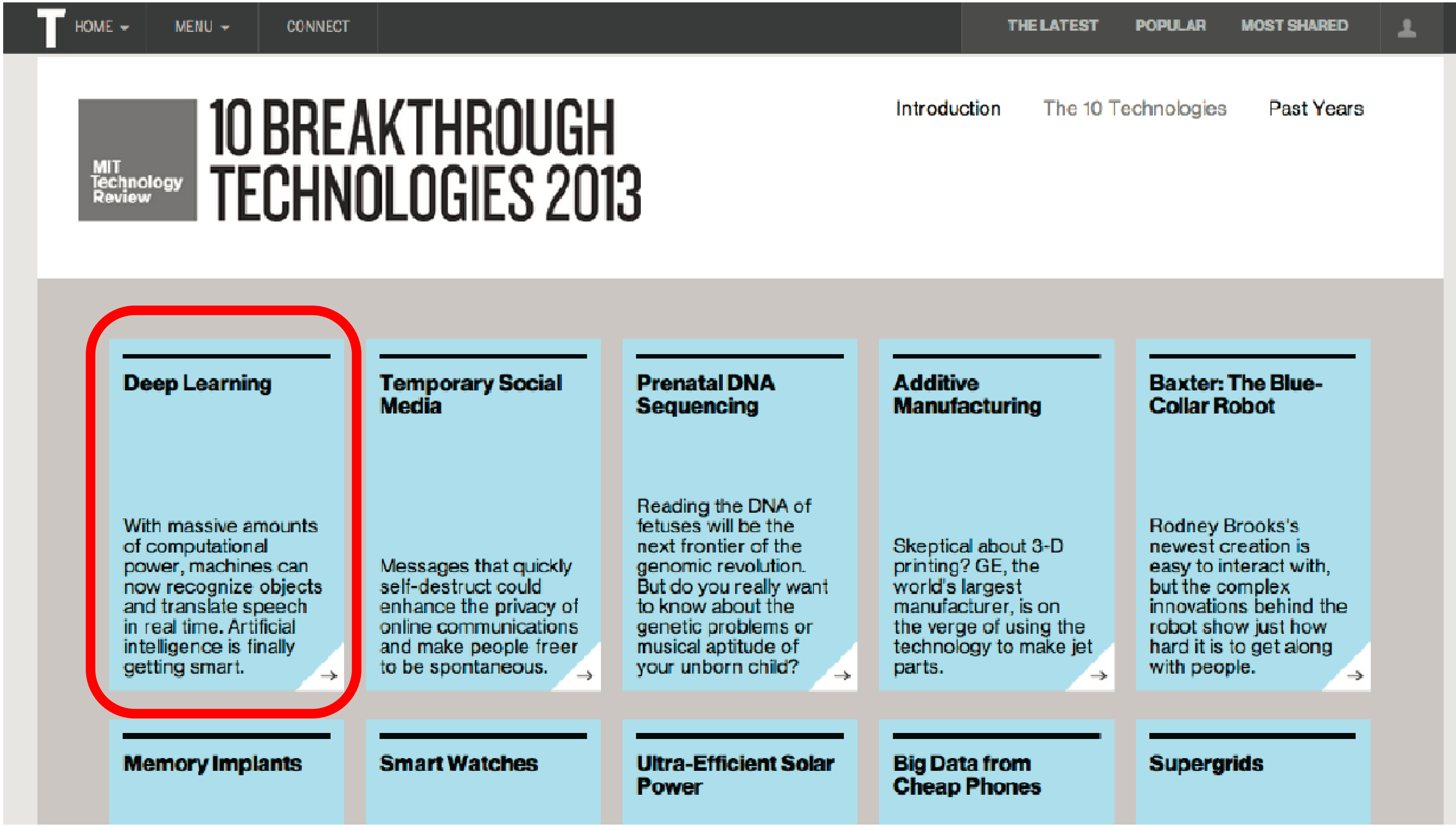






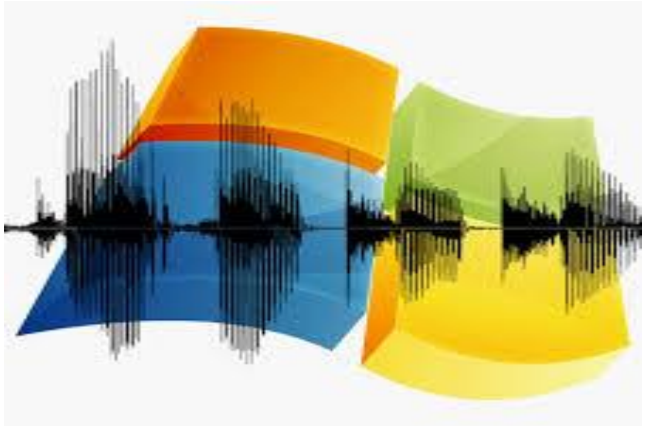
Deep Learning (Discussion)

Buzz…

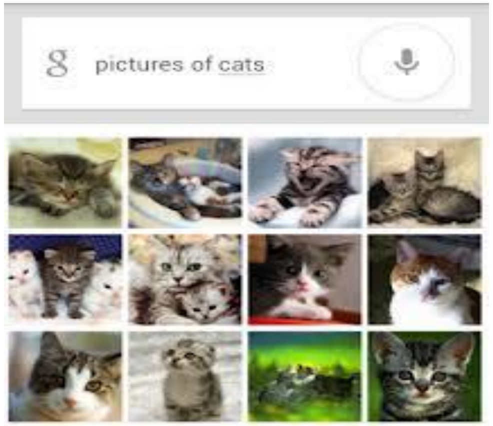


MIT Technology Review, April 23rd, 2013

Deep Learning Applications



Google Translate



Deep Learning - breakthrough in visual and speech recognition

Deep Mind (Learn to Play Game Go)

Classical Computer Vision Pipeline.

CV experts

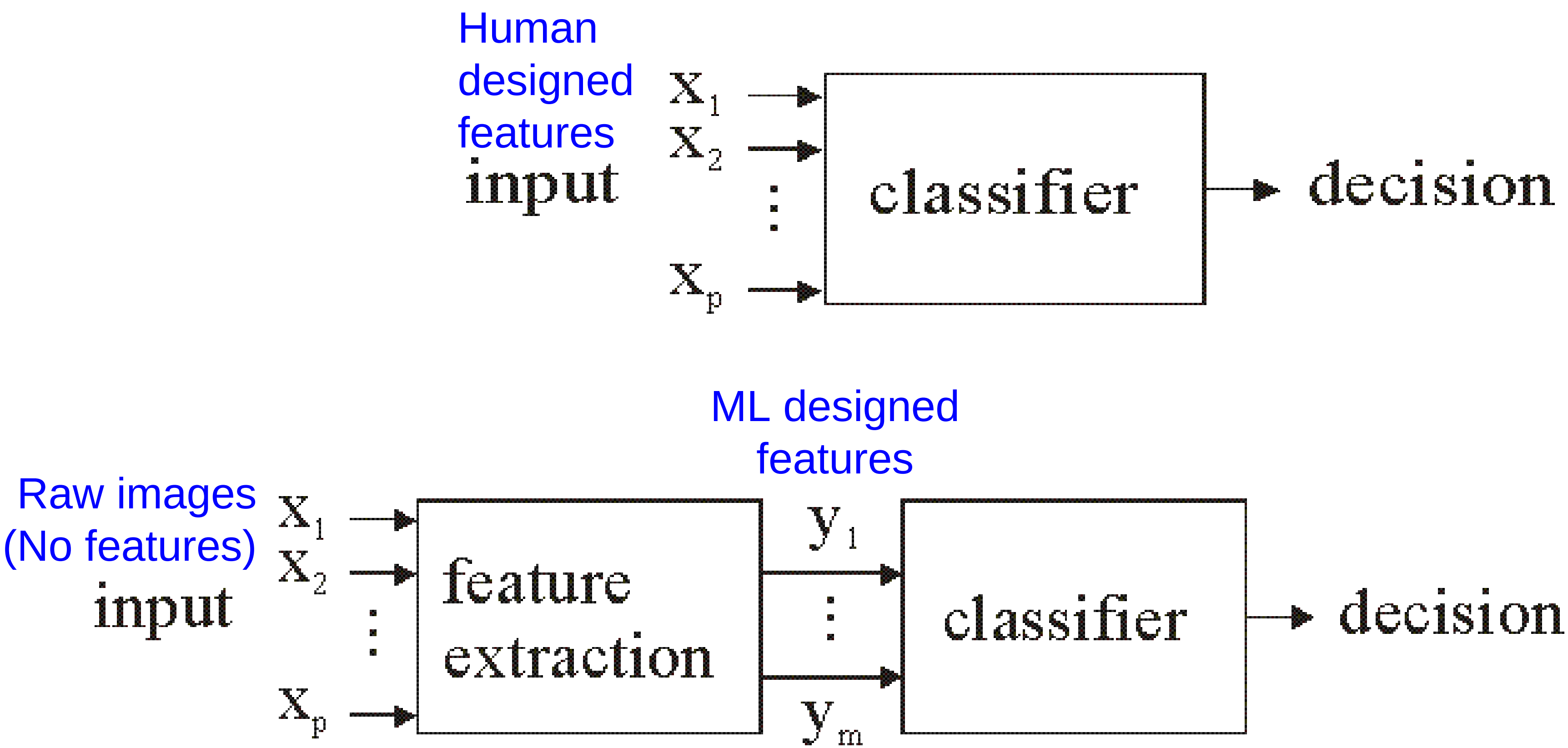
1. **Human select / develop features: SURF, HoG, SIFT, RIFT, …**
2. **Add on top of this Machine Learning for multi-class recognition and train classifier**

Feature Extraction: SIFT, HoG...

Detection, Classification Recognition

Classical CV feature definition is domain-specific and time-consuming

Vision: Classic vs. Deep-Learning



http://www.electronicsletters.com/papers/2001/0020/paper.asp

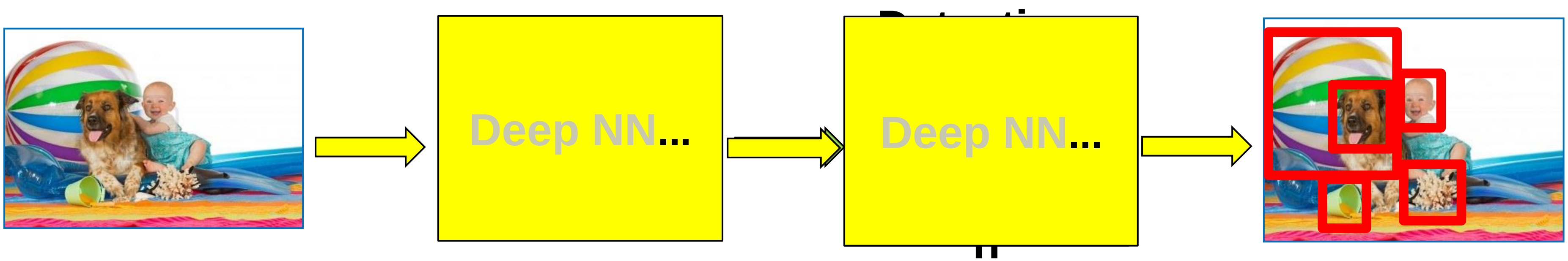
Deep Learning Vision Pipeline.

Deep Learning:

Build features automatically based on training data

Combine feature extraction and classification

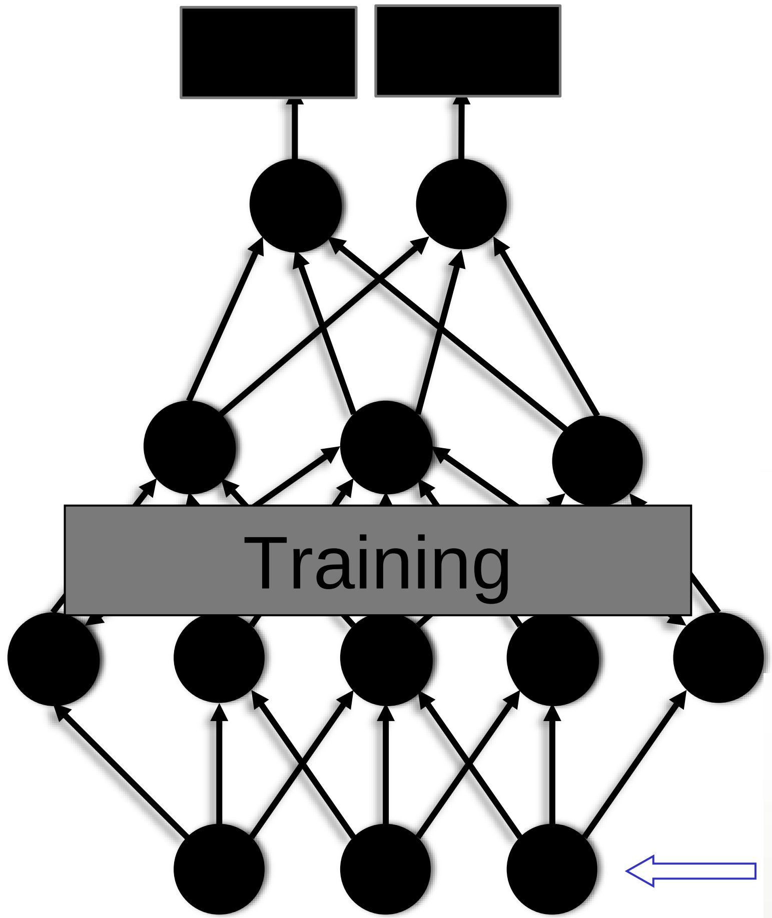
DL experts: define NN topology and train NN



Deep Learning promise:
train good feature automatically,
same method for different domain

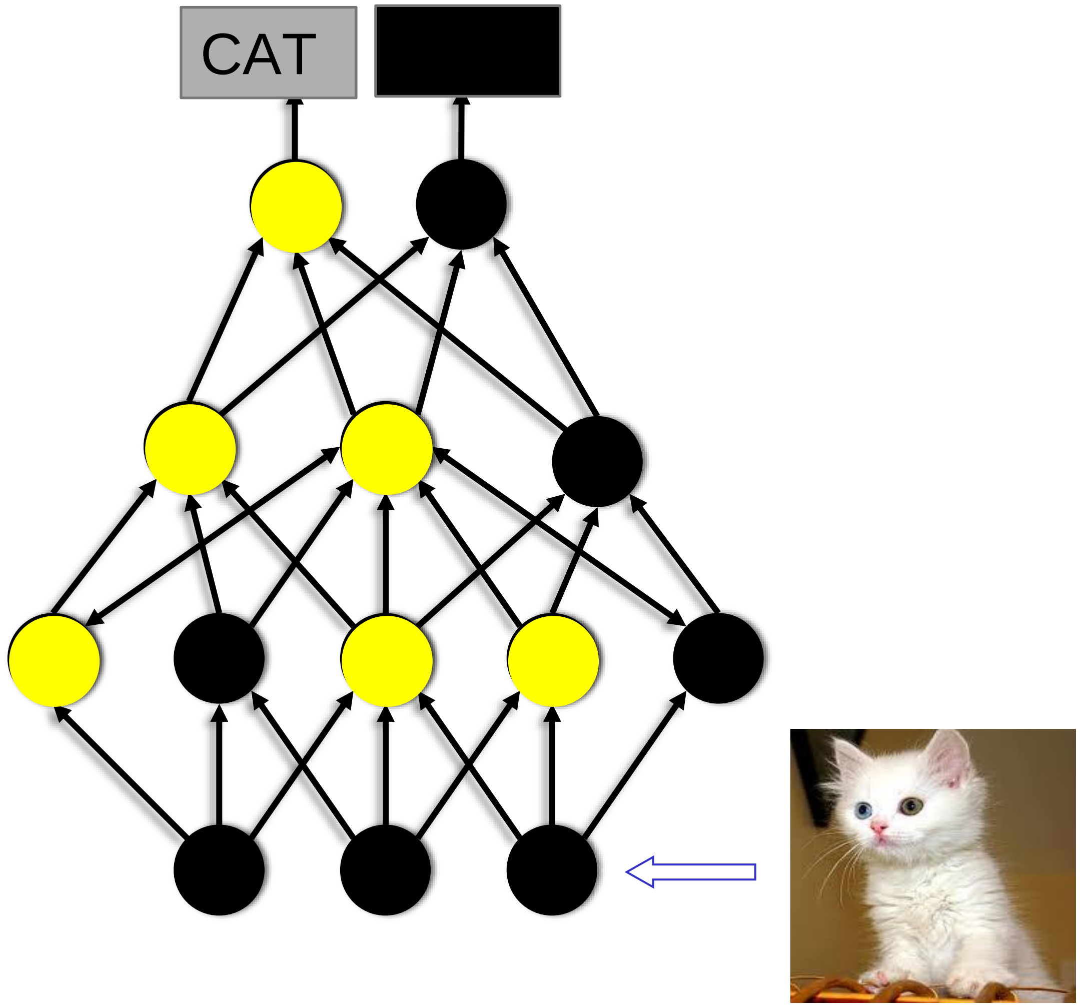
Deep Learning Basics

Deep Learning – is a set of machine learning algorithms based on multi-layer networks



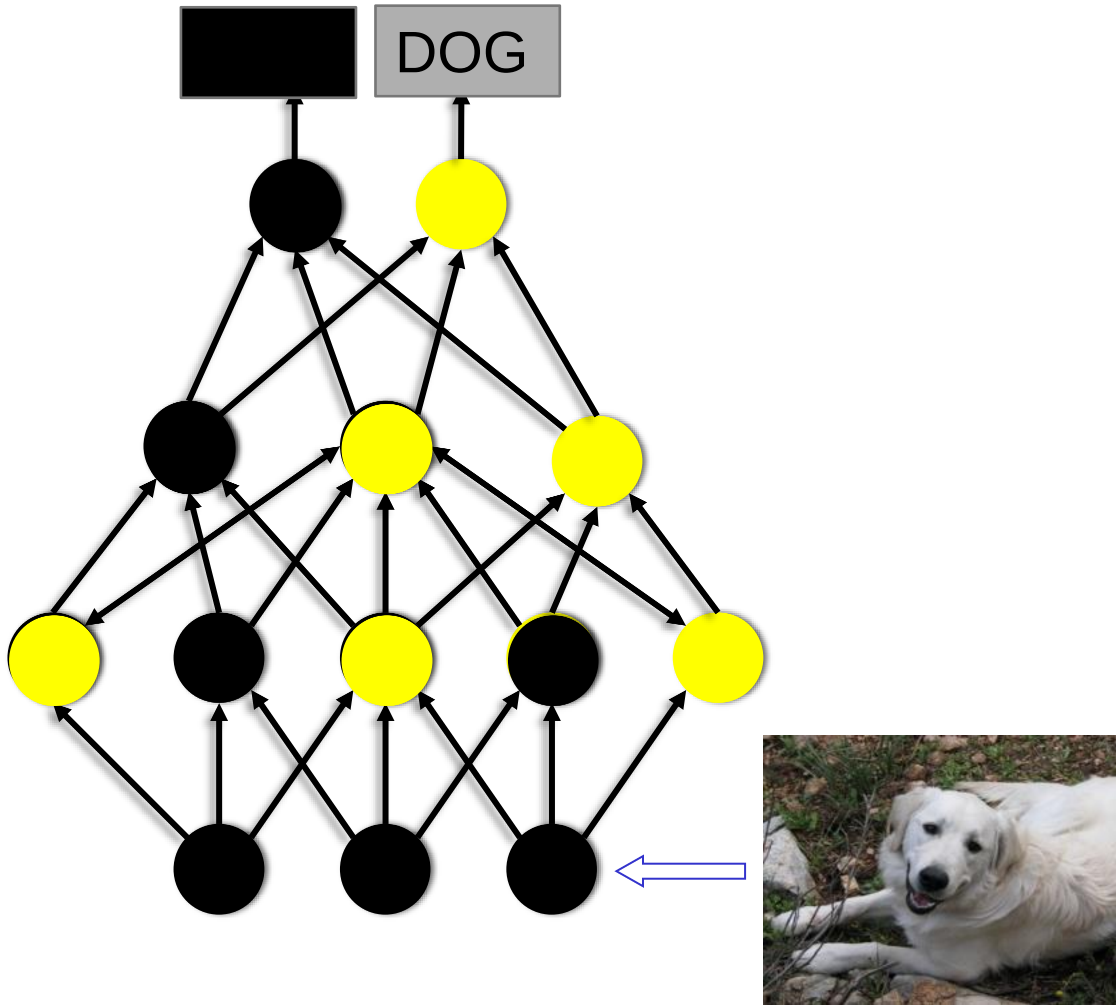
Deep Learning Basics

Deep Learning – is a set of machine learning algorithms based on multi-layer networks



Deep Learning Basics

Deep Learning – is a set of machine learning algorithms based on multi-layer networks



Deep Learning Taxonomy

Supervised:

- Convolutional NN (LeCun)
- Recurrent Neural nets (Schmidhuber)

Unsupervised

- Deep Belief Nets / Stacked RBMs (Hinton)
- Stacked denoising autoencoders (Bengio)
- Sparse AutoEncoders (LeCun, A. Ng,)

Google's TensorFlow Algorithm (highly recommended)

TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems

(Preliminary White Paper, November 9, 2015)

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng

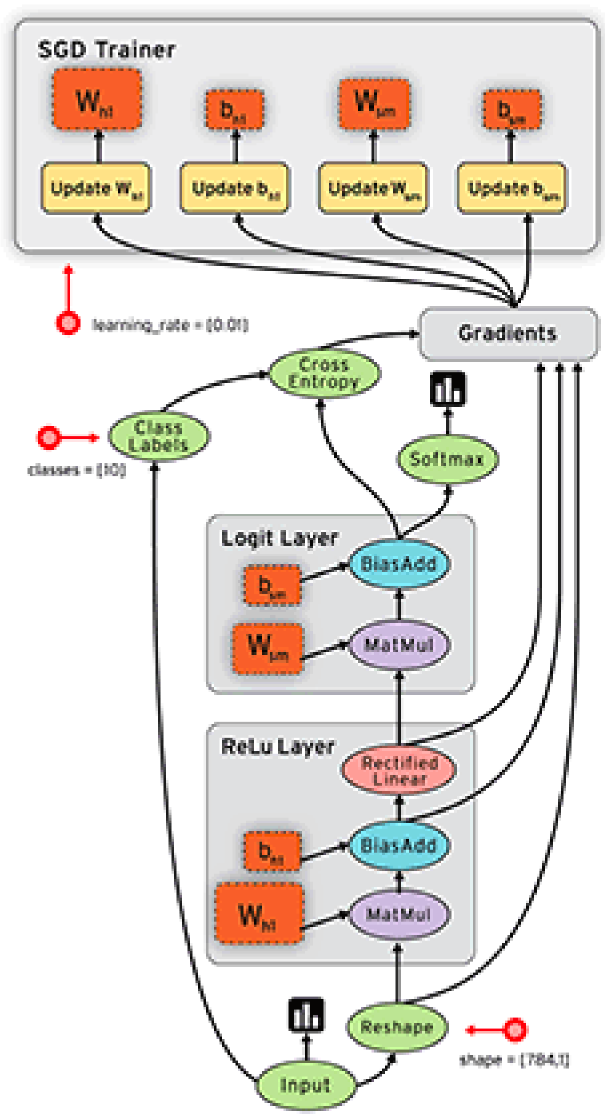
Google Research*

Abstract

TensorFlow [1] is an interface for expressing machine learning algorithms, and an implementation for executing such algorithms. A computation expressed using TensorFlow can be executed with little or no change on a wide variety of heterogeneous systems, ranging from mobile devices such as phones and tablets up to large-scale distributed systems of hundreds of machines and thousands of computational devices such as GPU cards. The system is flexible and can be used to express

sequence prediction [47], move selection for Go [34], pedestrian detection [2], reinforcement learning [38], and other areas [17, 5]. In addition, often in close collaboration with the Google Brain team, more than 50 teams at Google and other Alphabet companies have deployed deep neural networks using DistBelief in a wide variety of products, including Google Search [11], our advertising products, our speech recognition systems [50, 6, 46], Google Photos [43], Google Maps and StreetView [19],

TensorFlow Graph



TensorFlow Highlights

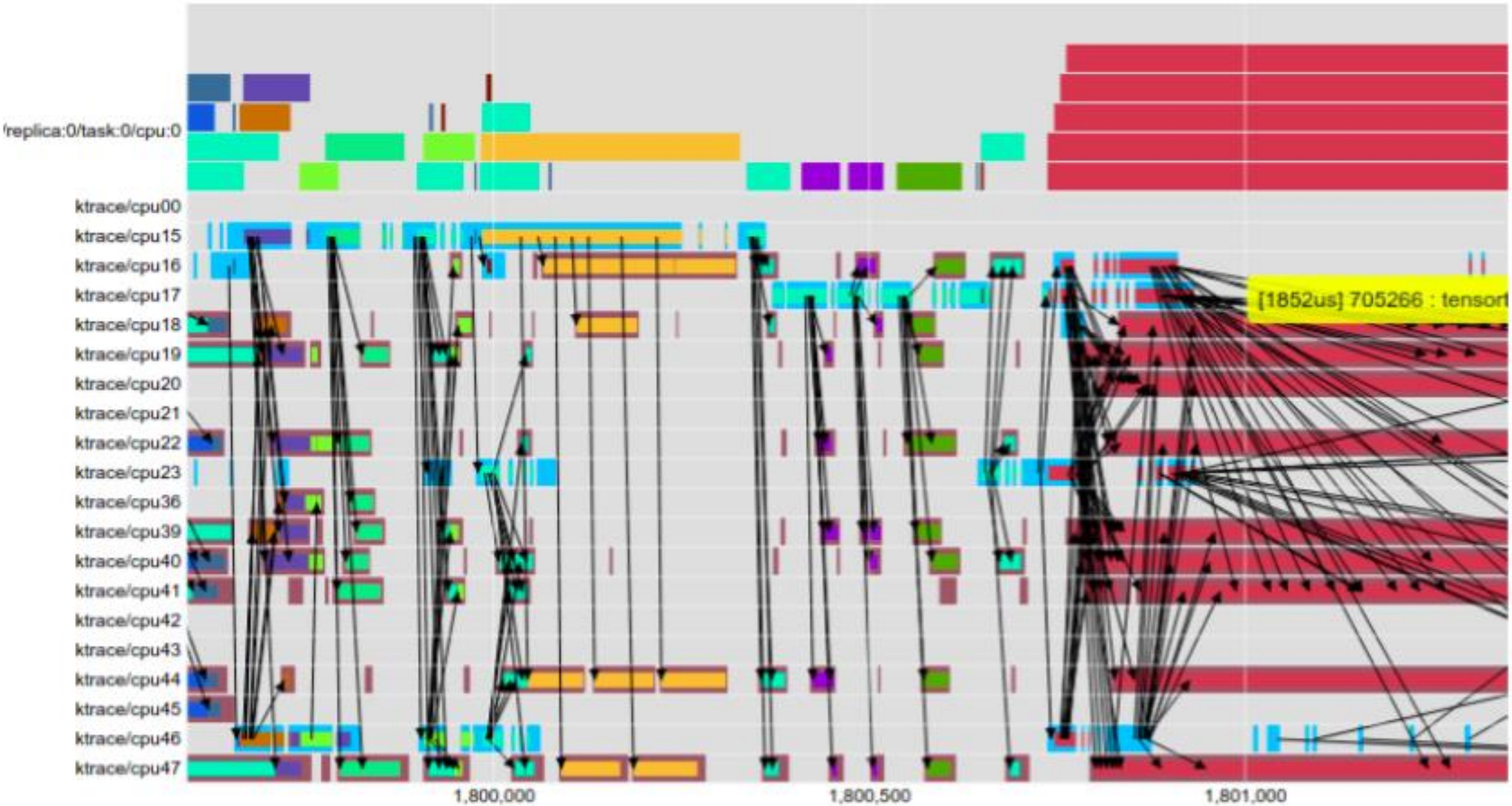


Figure 12: EEG visualization of multi-threaded CPU operations (x-axis is time in μs).