ELEC/COMP 576: Introduction to Deep Machine Learning Lecture 2

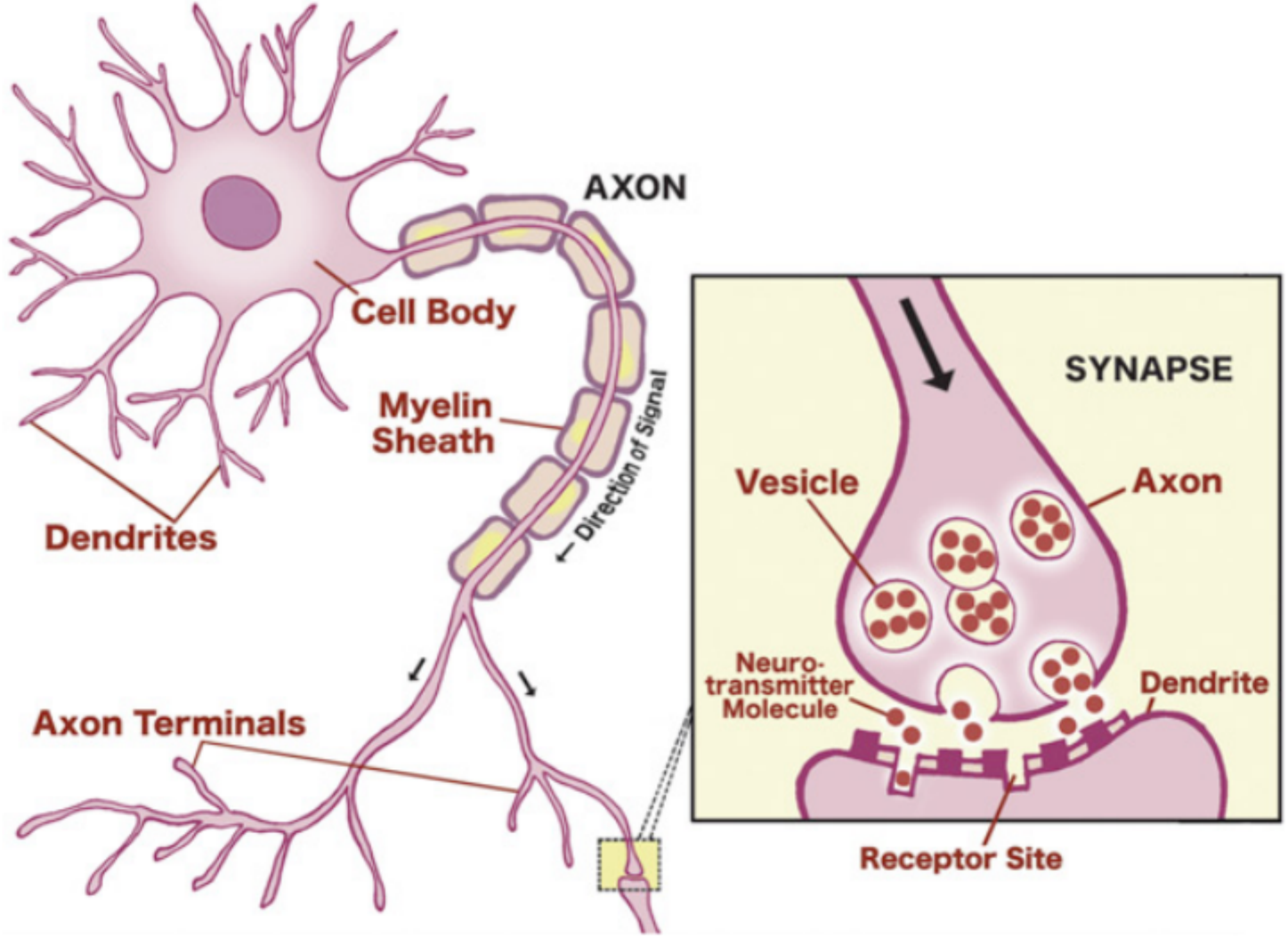
Ankit B. Patel
Baylor College of Medicine (Neuroscience Dept.)
Rice University (ECE Dept.)

A Brief History of Neural Networks

History

Time	Event
1943	McCulloch + Pitts introduce a simplified mathematical model of neurons as computing ANDs and ORs: capable of basic Logic but no Learning yet. They prove theorems about **expressive power**.
1949	Hebb introduces the first neurobiological **learning rule**: "*Neurons that fire together, wire together.*"
1950s	Computers are in their infancy. *MADALINE*: The first NN for commercial use: designed for adaptive filtering of echoes in phone lines (1959). Still in use today!!
1962	Rosenblatt introduces the ***Perceptron***. It learns! Hubel + Wiesel describe simple and complex cells in visual area V1 (inspiration for later NNs: S-->**template matching** for pattern specificity and C-->**pooling** for robustness to nuisances).
1969-1981	Minsky and Papert deal "deathblow" to Perceptrons by showing they can't learn XOR. Later, others showed this wasnt true for deep NNs. "NN Winter": excessive hype and outrageous claims lead to dying interest/funding.

An Example Neuron



How do neurons communicate?

- Animations of excitatory and inhibitory neuron:
- https://nba.uth.tmc.edu/neuroscience/s1/introduction.html

McCulloch-Pitts Neurons (1943)

LOGICAL CALCULUS OF IDEAS IMMANENT IN NERVOUS ACTIVITY

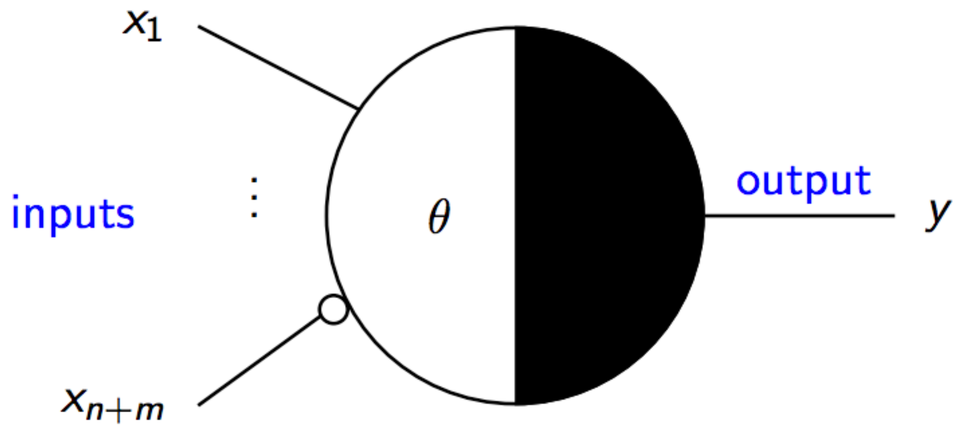
Warren S. McCulloch Walter H. Pitts

Bulletin of Mathematical Biophysics 5:115-133 (1943)

Assumptions (drawn from Empirical Observations)

1. The activity of the neuron is an ‘‘all-or-none’’ process.
2. A certain fixed number of synapses must be excited within the period of latent addition in order to excite a neuron at any time, and this number is independent of previous activity and position of the neuron.
3. The only significant delay within the nervous system is synaptic delay.
4. The activity of any inhibitory synapse absolutely prevents excitation of the neuron at that time.
5. The structure of the net does not change with time.

McCulloch-Pitts Neurons



Stefan Droste

- all signals binary ($\in \{0, 1\}$)
- threshold $\theta \in \mathbb{N}_0$
- n excitatory inputs $x_1, \ldots, x_n$
- m inhibitory inputs $x_{n+1}, \ldots, x_{n+m}$
- one output y (with unrestricted fan-out)

$$y(x_1, \ldots, x_{n+m}, \theta) = \begin{cases} 1 & \text{if } \sum_{i=1}^{n} x_i \geq \theta \text{ and } \sum_{i=n+1}^{m} x_i = 0 \\ 0 & \text{if } \sum_{i=1}^{n} x_i < \theta \text{ or } \sum_{i=n+1}^{m} x_i > 0 \end{cases}$$

McCulloch-Pitts Nets

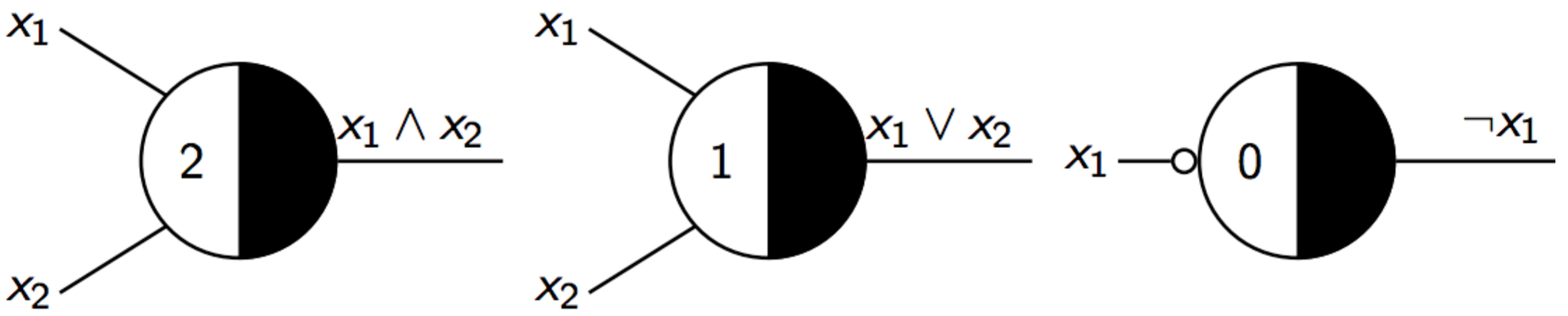
A McCulloch-Pitts Net is a directed graph G with McCulloch-Pitts neurons as nodes and edges marked as either excitatory or inhibitory.
If G is acyclic it is called a feed-forward net.
Otherwise, G is called recursive net.

In recursive McCulloch-Pitts nets, each neuron computes an output in 1 time step.

Stefan Droste

Expressive Power of McCulloch-Pitts Nets

Feed-forward McCulloch-Pitts nets can compute any Boolean function $f : \{0,1\}^n \to \{0,1\}^m$.



Stefan Droste

Recursive McCulloch-Pitts nets can simulate any deterministic finite automaton (DFA).

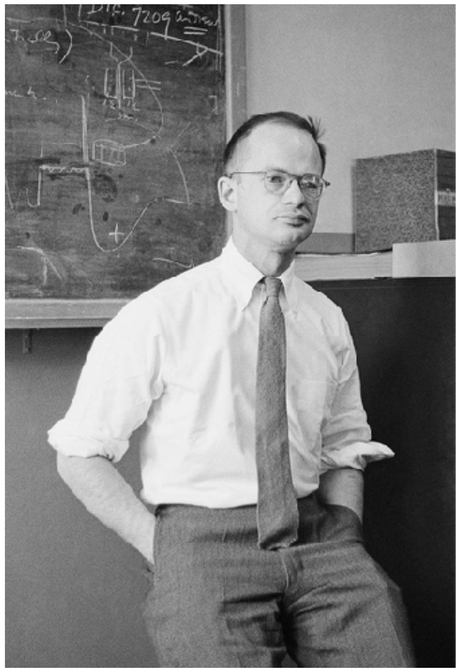
Why is it important?

- A formalism whose refinement and generalization led to the notion of “finite automata”
- A technique that inspired the notion of logic design
- The first use of computation to address the mind-body problem
- The first modern computational theory of mind and brain

Aside: The Tragic Story of Walter Pitts

- **Must read:** http://nautil.us/issue/21/information/the-man-who-tried-to-redeem-the-world-with-logic

- The downfall of universal expressive power: it does not provide any constraint on the microscopic mechanisms by which a NN computes.



Question:

What are the problems/limitations of expressive power? (1-2 min)

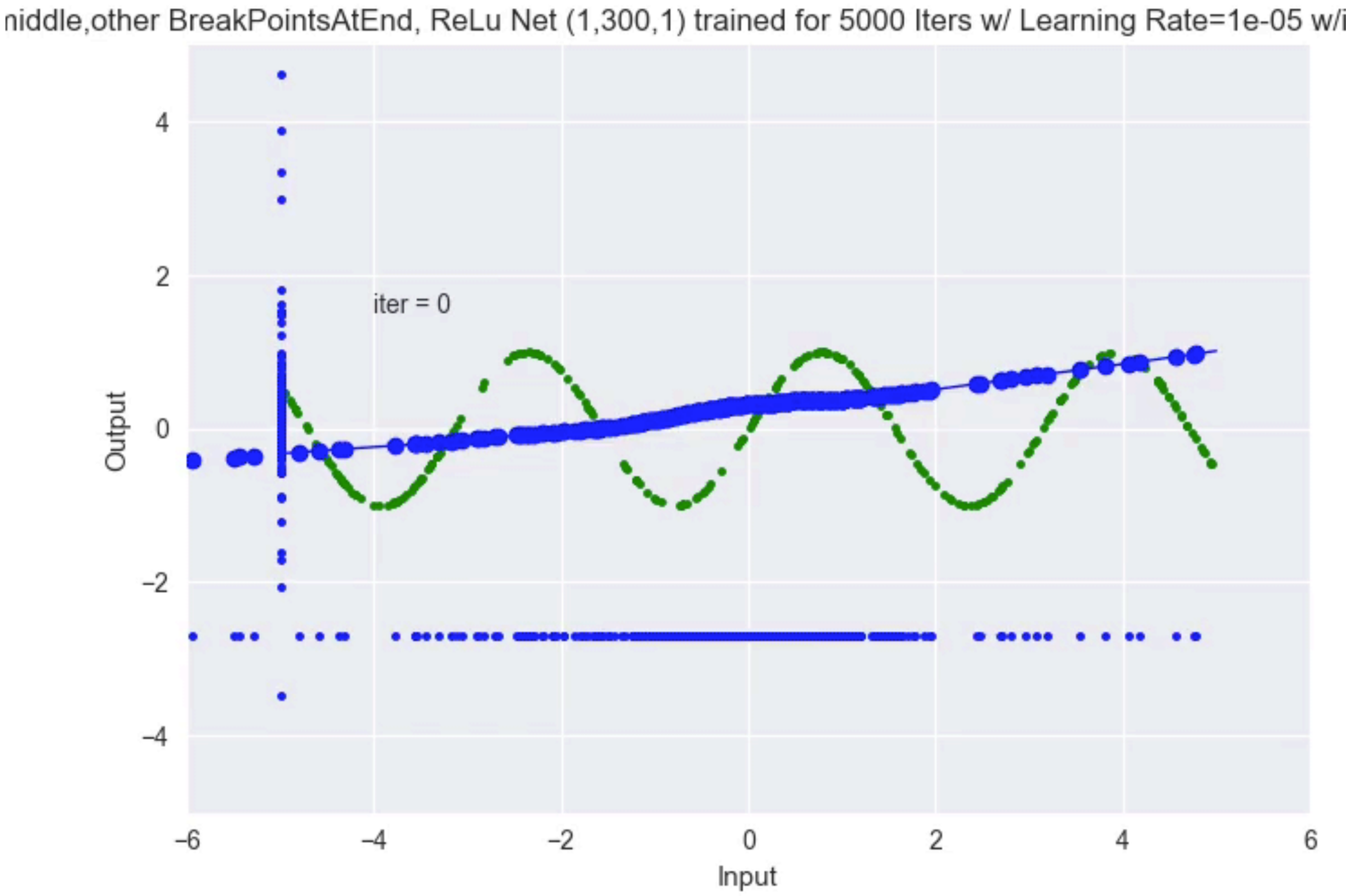
Question:

What are the problems/limitations of expressive power? (1-2 min)

Answer:

Expressability does not imply Learnability

Expressability vs. Learnability

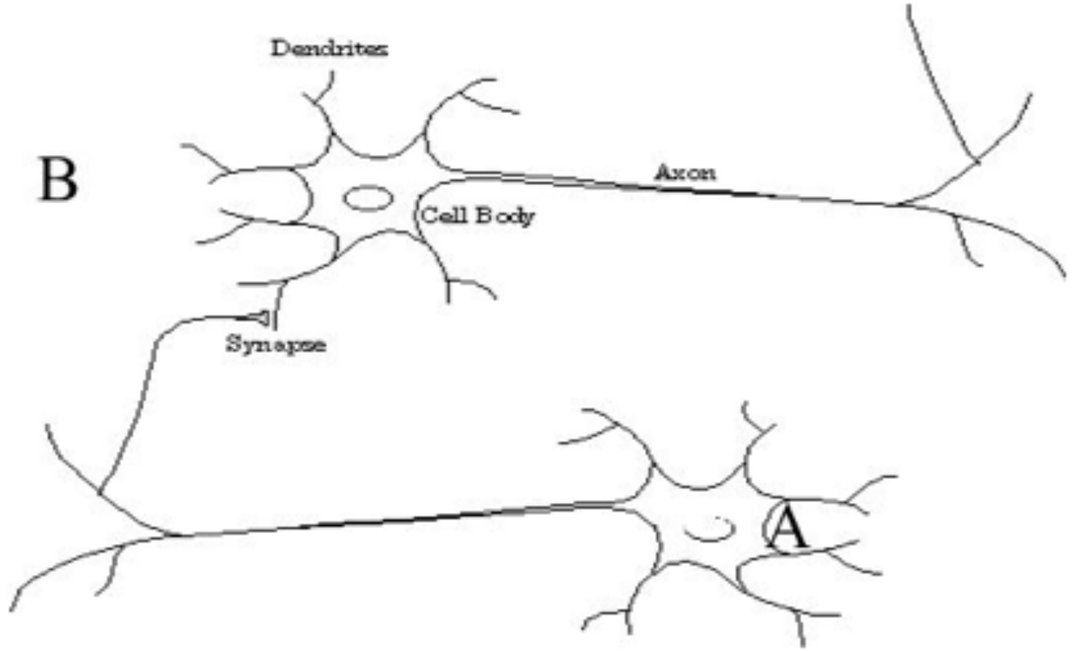


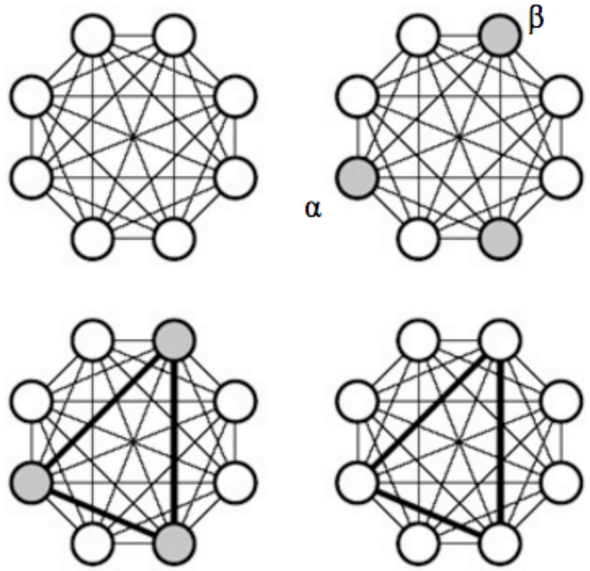
Can easily express function
But difficult to learn/optimize (not enough breakpoints nearby
and not able to move them)

How can Neurons learn? Hebb's Postulate

"When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased."

D. O. Hebb, 1949





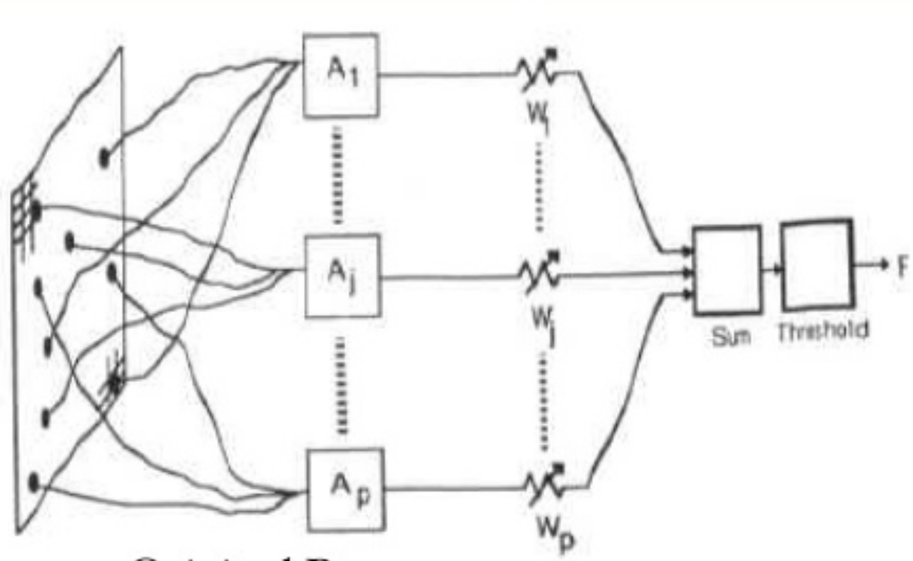
How do neurons process visual input?

- Hubel and Wiesel's incredible discovery (1962):
  - https://www.youtube.com/watch?v=IOHayh06LJ4
- Awarded Nobel Prize in Physiology and Medicine

The Perceptron (Rosenblatt 1957)

$$f(x) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ 0 & \text{else} \end{cases}$$

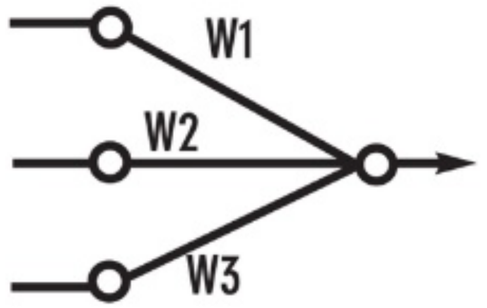
- First architecture to have a *learning* algorithm:



Original Perceptron

(From Perceptrons by M. L Minsky and S. Papert, 1969, Cambridge, MA: MIT Press. Copyright 1969 by MIT Press.

Frank Rosenblatt
(1928-1971)



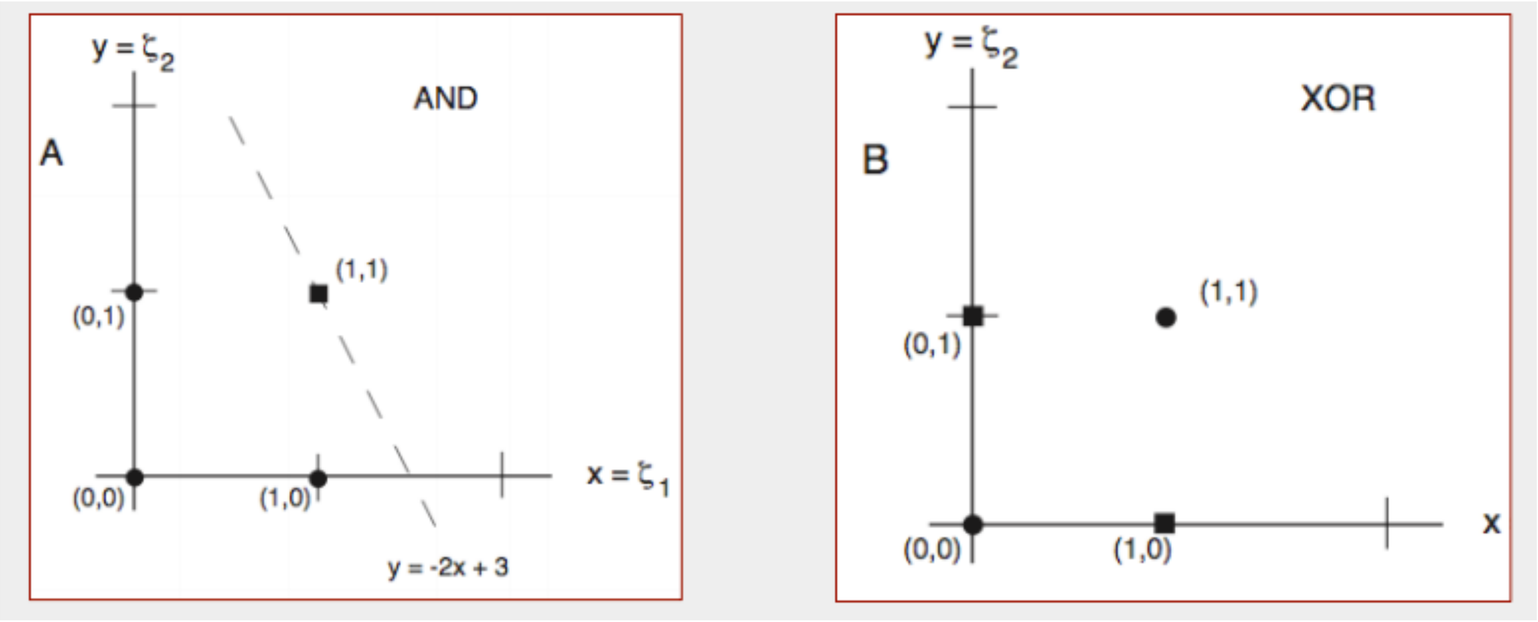
Simplified model:

$$y_j(t) = f[\mathbf{w}(t) \cdot \mathbf{x}_j]$$
$$= f[w_0(t)x_{j,0} + w_1(t)x_{j,1} + w_2(t)x_{j,2} + \cdots + w_n(t)x_{j,n}]$$

$w_i(t+1) = w_i(t) + r \cdot (d_j - y_j(t))x_{j,i}$, for all features $0 \leq i \leq n$, r is the learning rate.

Minsky & Papert Deal a “Deathblow” to the Perceptron: The XOR Problem

Linear Separable or Not?



Question:

How can you prove that the XOR Problem is not linearly separable? (4 min)

Answer: Proof by Contradiction

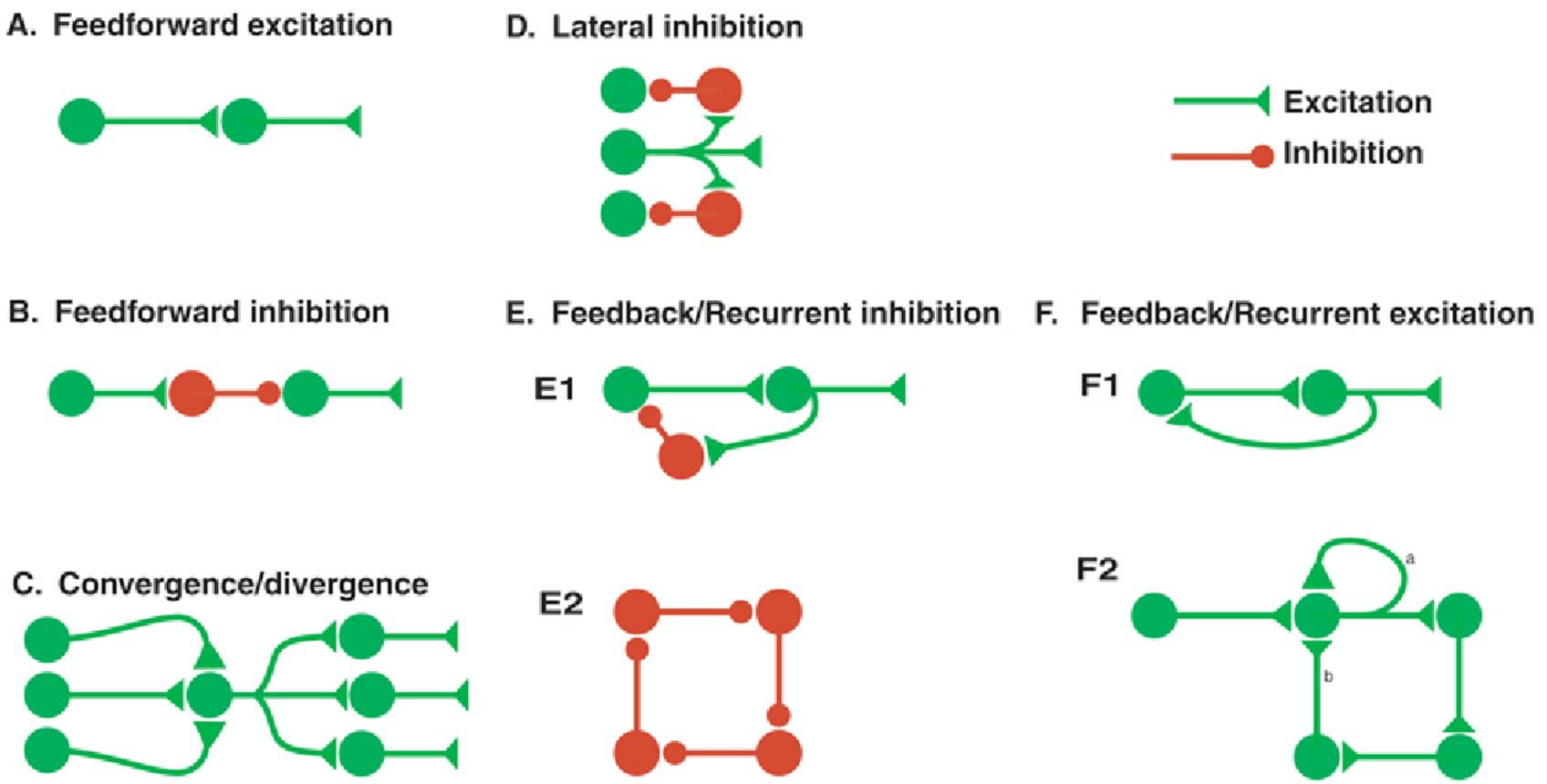
$$w_1 x + w_2 y = w_1 0 + w_2 0 < \theta$$
$$w_1 x + w_2 y = w_1 0 + w_2 1 \geq \theta$$
$$w_1 x + w_2 y = w_1 1 + w_2 0 \geq \theta$$

Greater than zero.

$$w_1 x + w_2 y = w_1 1 + w_2 1 < \theta$$

Impossible, given above.

Final Project Idea: MicroNetwork Motifs



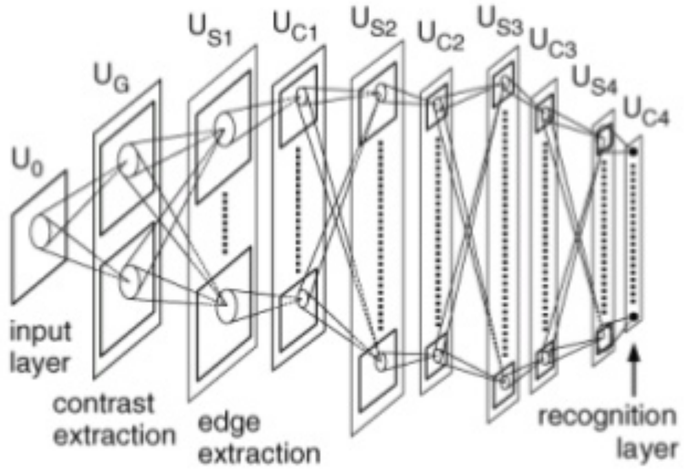
- https://nba.uth.tmc.edu/neuroscience/s1/introduction.html

History

1970 +- 10 years	**Backpropagation** (Gradient Descent for nested functions via Chain Rule) explored in various contexts (both NN-specific and general).
1979	Fukushima introduces the *Neocognitron*. It foreshadows current deep NNs: **convolutional** layers, **weight replication**, and WTA-**subsampling**. However its unsupervised.
1982	**Hopfield Nets**: RNNs with binary units that can serve as an content-addressable associative memory (given a partial/wrong pattern, it will complete/correct it to the most similar memory). Revival of interest ensued.
1986	Werbos, Rumelhart + Hinton Williams revive Backprop for NNs. Hinton and Smolensky introduce **(Restricted) Boltzmann machines (R)BMs**.
1989	LeCun applies Backprop to Fukushima's Neocognitron to do supervised learning. This is the first incarnation of modern **convolutional neural nets (CNNs)**. Subsequently used by US Post Office for address reading.
1990	Baird introduces the idea of **data augmentation**: adding transformed/deformed versions of original input data to increase size of training dataset.
1991-5	Fundamental Problem of DL: **vanishing/exploding gradients** plague deep FFNNs and RNNs. Hochreiter + Schmidhuber introduce LSTM RNNs (1995-7) to address this. LSTMs are critical to solving problems requiring a real memory (e.g. widely spaced events in time are important.)

Neocognitron: Precursor to Modern Convnets

Fukushima (1980). Hierarchical multilayered neural network

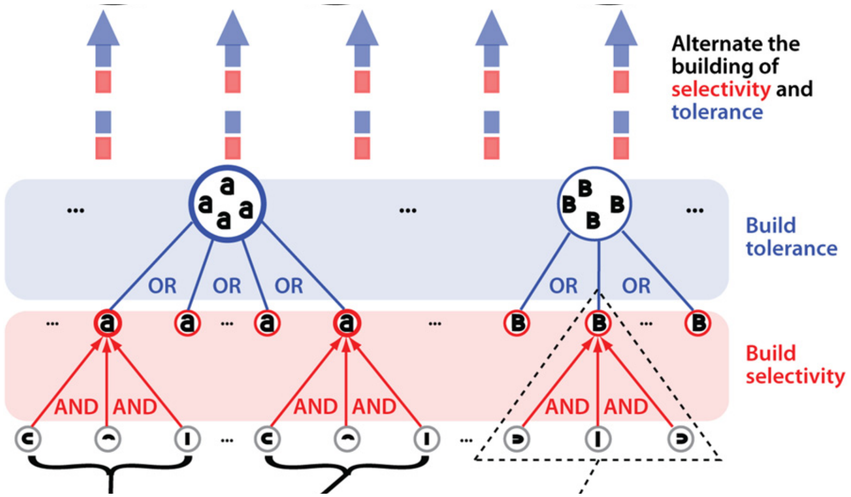


S-cells work as feature-extracting cells. They resemble simple cells of the primary visual cortex in their response.

C-cells, which resembles complex cells in the visual cortex, are inserted in the network to allow for positional errors in the features of the stimulus. The input connections of C-cells, which come from S-cells of the preceding layer, are fixed and invariable. Each C-cell receives excitatory input connections from a group of S-cells that extract the same feature, but from slightly different positions. The C-cell responds if at least one of these S-cells yield an output.

Key Idea: Alternating Selectivity and Invariance

- Hubel and Wiesel's discovery of simple/complex cells and their special properties of *selectivity* and *tolerance/invariance*



DiCarlo, J. J. et al. How does the brain solve visual object recognition? Neuron (2012).

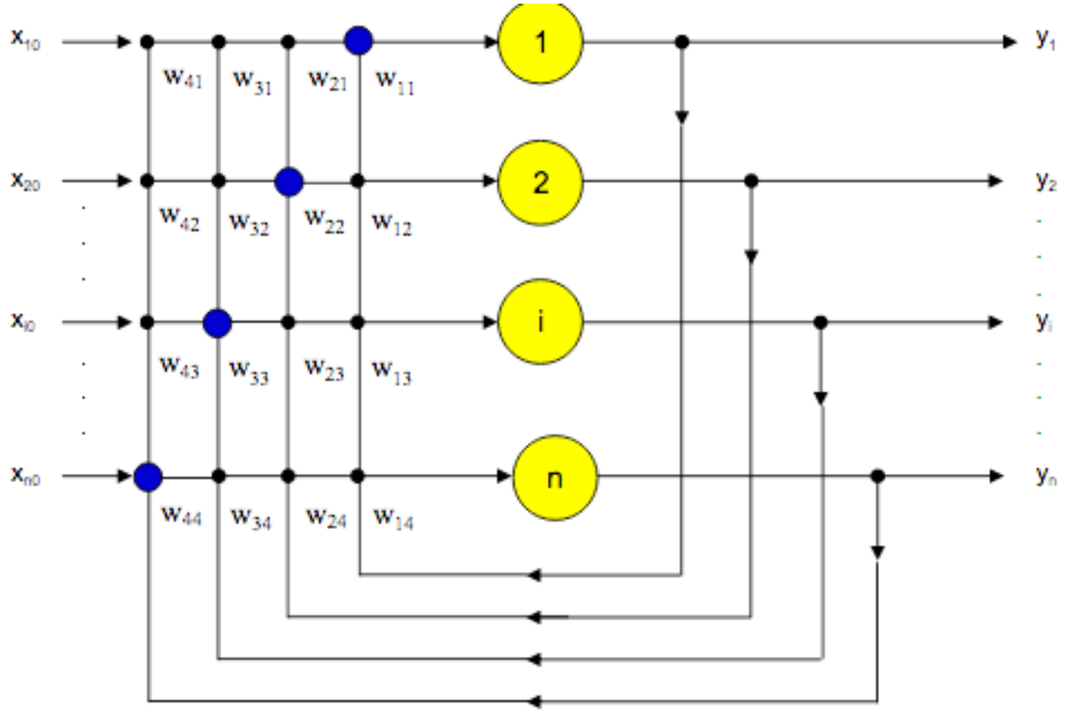
Key Inspiration from Neuroscience

Build up feature selectivity and tolerance over multiple layers in a hierarchy $\Rightarrow$
ML architectures: Neocognitron, HMAX, SIFT, and modern **Deep Convnets**

Recurrent Neural Networks: Hopfield Nets

- A feedback neural network has feedback loops from its outputs to its inputs. The presence of such loops has a profound impact on the learning capability of the network.
- After applying a new input, the network output is calculated and fed back to adjust the input. This process is repeated until the outcome becomes constant.
- John Hopfield (1982)
  - Associative Memory via artificial neural networks
  - Optimisation

Hopfield Nets: Mathematical Definition



$$y_j(t) = \mathrm{sgn}(x_j(t)); \quad \mathrm{sgn} = \begin{cases} +1 & \text{if } x > 0 \\ -1 & \text{if } x < 0 \end{cases}$$

It is a dynamic system:
$x(0) \rightarrow y(0) \rightarrow x(1) \rightarrow y(1) \ldots \rightarrow y^*$

$$x_j(t) = \sum_{i=1}^{n} w_{ij} y_i(t-1)$$

Hopfield Nets: Engineering the Attractor

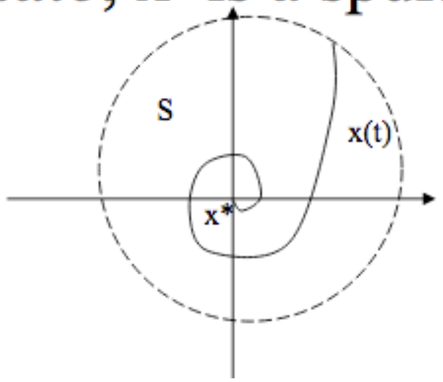
- Attractor

  If a state x(t) in a region S and $t \rightarrow \infty$,, $x(t) \rightarrow x^*$

  S is the attractive region.

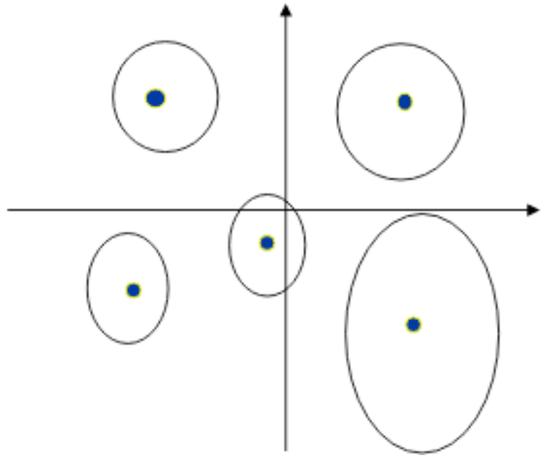
  if x^*=*desired state*, x^* is an attractor

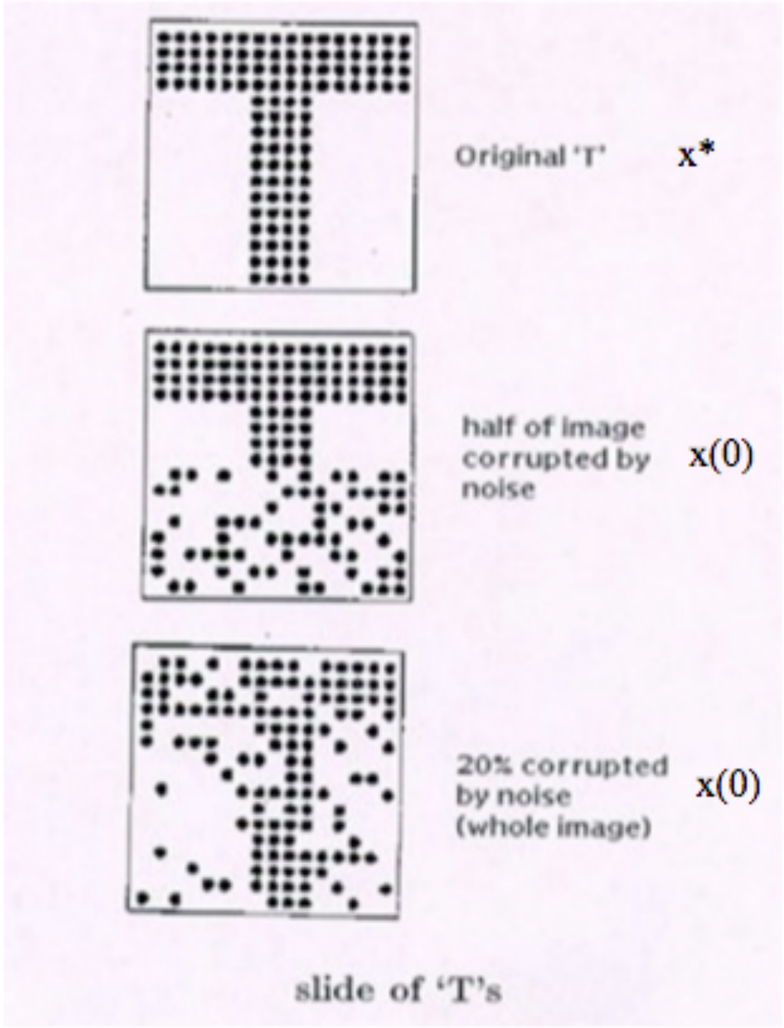
  if $x^* \neq$*desired state*, x^* is a spurious attractor.



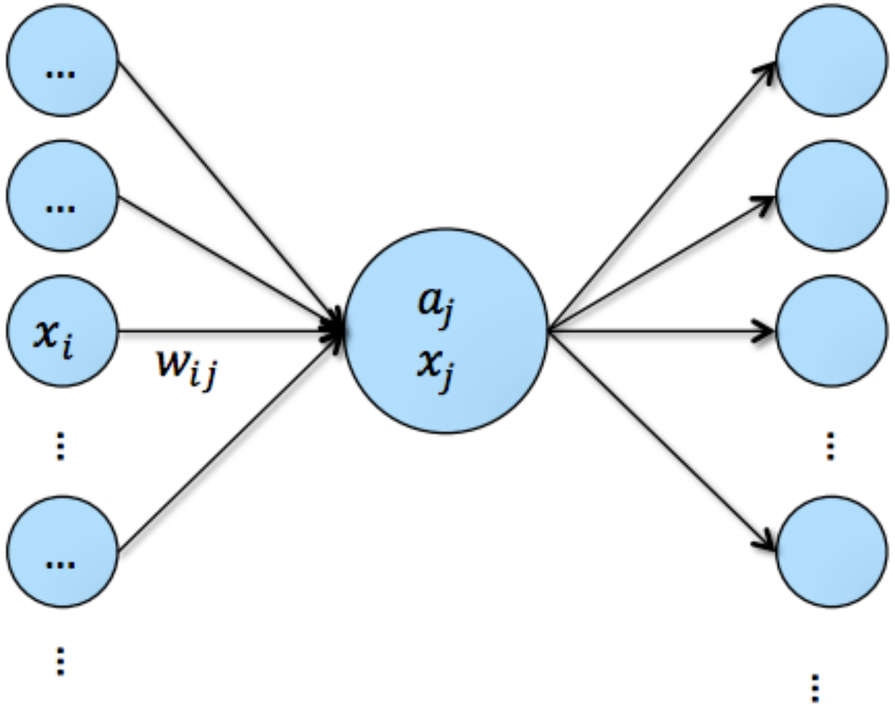
Hopfield Nets as Associative Memory

- Nature of associative memory
  - part of information given
  - the rest of the pattern is recalled
- Hopfiled networks can be used as associative memory.
  - Design weight W so that X^*=the memorised patterns.
  - Can store more than one. Capacity increases





How to Learn NNs?
The Backpropagation Algorithm (1960-86)



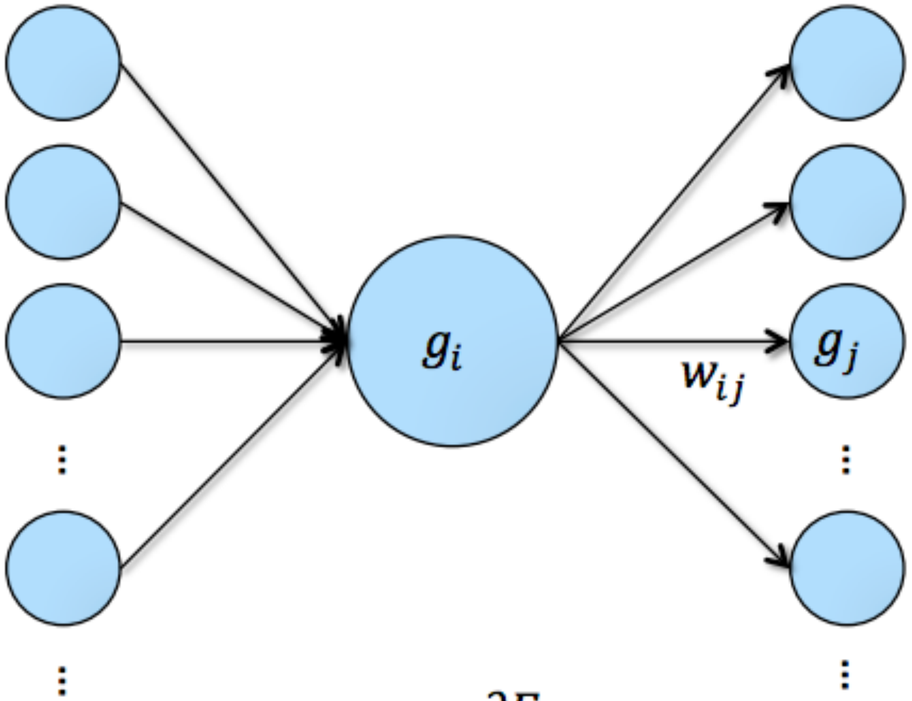
Net Input

$$a_j = \sum_{i \in Pre(j)} w_{ij}\, x_i$$

Activation + Nonlinearity

$$x_j = f(a_j)$$

How to Learn NNs?
The Backpropagation Algorithm (1960-86)



$$g_\bullet = \frac{\partial E}{\partial a_\bullet}$$

Chain rule

$$g_i = f'(a_i) \sum_{j \in Post(i)} w_{ij}\, g_j$$

$$\frac{\partial E}{\partial w_{ij}} = x_i\, g_j$$

The History of the Backpropagation Algorithm (1960-86)

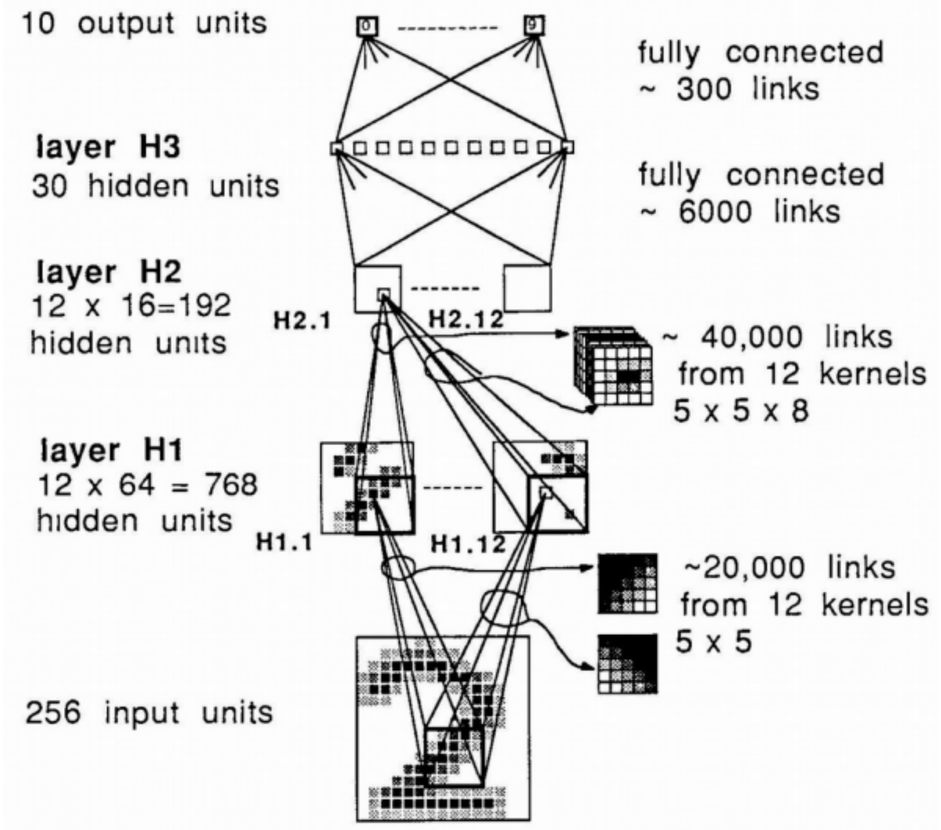
- Introduced in **Control Theory**, via Dynamic Programming [Henrey J. Kelley (1960) & Arthur Bryson (1961)]
- Simpler derivation using **Chain Rule** [Stephen Dreyfus (1962)]
- General method for **Automatic Differentiation** [Seppo Linnainamaa (1970)]
- Using Backprop to estimating parameters of controllers with objective of minimizing error [Stuart Dreyfus (1973)]
- Backprop brought into NN world [Paul Werbos (1974)]
- Used BP to learn representations in hidden layers of NNs [Rumelhart, Hinton & Williams (1986)]

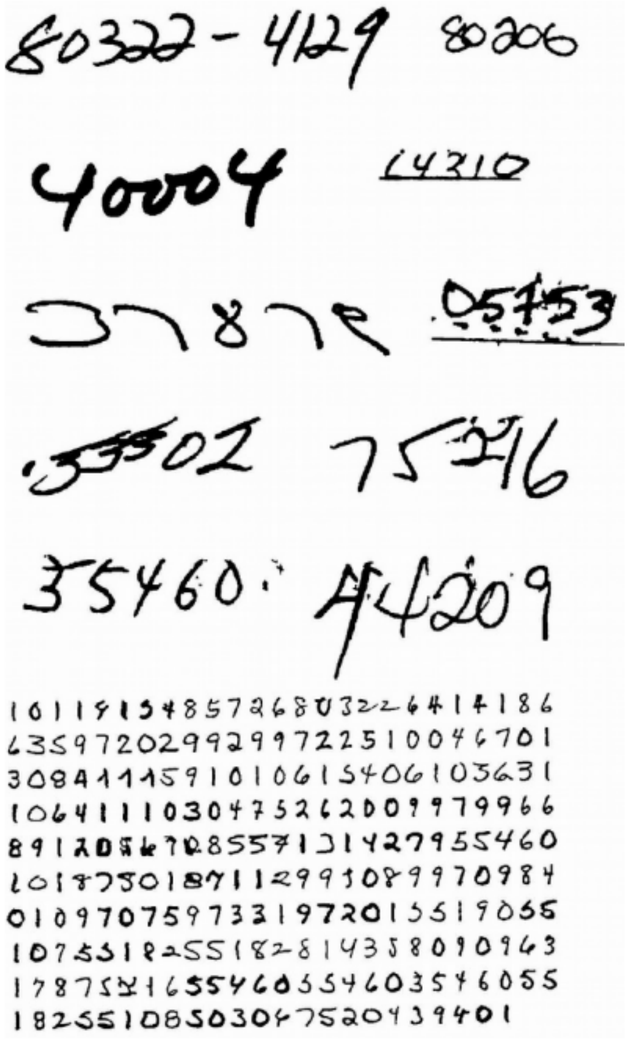
Solving Digit Recognition for the US Post Office: (Yann Lecun 1989)

Trained with Backprop.

USPS Zipcode digits: 7300 training, 2000 test.

Convolution with stride. No separate pooling.

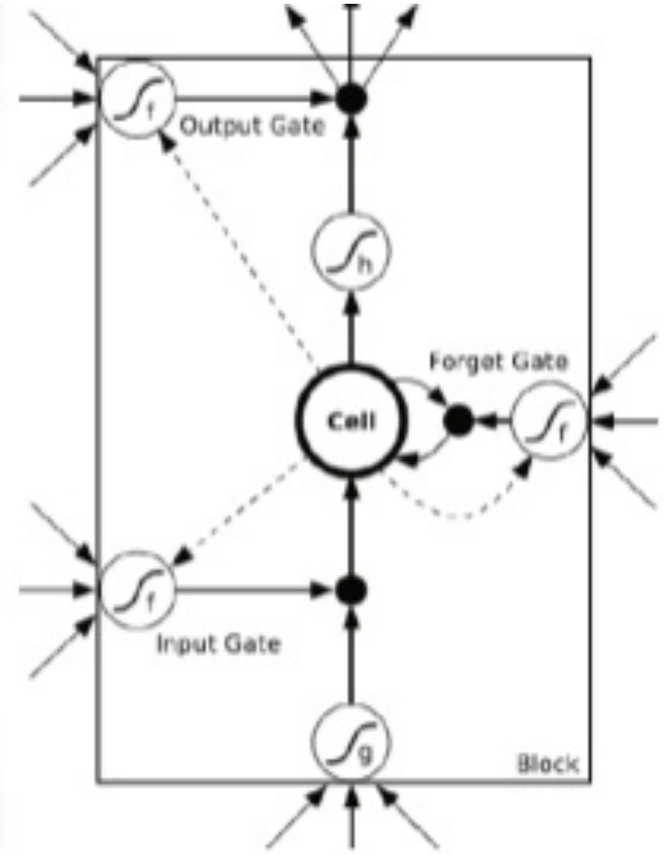




Long Short-Term Memory Recurrent Neural Networks (Hochreiter & Schmidhuber, 1992)

- Input Gate
- Forget Gate
- Cell
- Output Gate
- Hidden state output

$$
\begin{aligned}
i_t &= \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i\right) \\
f_t &= \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right) \\
c_t &= f_t c_{t-1} + i_t \tanh\left(W_{xc}x_t + W_{hc}h_{t-1} + b_c\right) \\
o_t &= \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o\right) \\
h_t &= o_t \tanh(c_t)
\end{aligned}
$$



History

1992	Weng introduces the *Cresceptron* and **Max-Pooling**. Critical for current DNNs. **Wavelets** and **Multi-resolution** analysis of images are introduced (but no learning).
1999-2004	Riesenhuber Poggio introduce **HMAX**: a hierarchical model of visual cortex. Lowe introduces **SIFT**, **scale-invariant feature transform**, that is partially inspired by selectivity/invariance properties described in HMAX. SIFT is deep (multi-scale descriptor) but there is no learning.
2001	Breimann introduces **Random Decision Forests (RFs)**: a combination of **Bagging** and randomly dropping features in order to prevent overfitting that was so common in decision trees. RFs are employed in many modern day image segmentation tasks in Medical Imaging and for pose estimation in 3D Gaming (Kinect). They also do "unreasonably" well in many ML competitions.
2006-7	Hinton and Salaktudinov use unsupervised pre-training + supervised fine-tuning to train Deep **RBMs**, setting new records on MNIST.

History

2006-7	Hinton and Salaktudinov use unsupervised pre-training + supervised fine-tuning to train Deep **RBMs**, setting new records on MNIST.
2006-10	**GPU** implementations of BP that are up to 50x faster than CPU are introduced, and can process lots more data in the same time. Emphasized that advances in hardware and Big Data may be more important than specific NN architecture.
2011	Martens and Sutskever introduce *Hessian-Free Optimization*: a training algorithm for RNNs that can alleviate vanishing gradient problem.
2012 - present	Krizhevsky and Hinton show that CNNs can win ImageNet competition. A resurgence of interest in DL: Lots more Contests and Benchmarks advanced. And the hype is back as well! A new *Theory of Deep Learning (developed here at Rice)* that shows **the probabilistic origin of deep CNNs and RFs**. Generative model ⇒ Probabilistic Inference ⇒ NN relaxation

Graphical Processing Units (GPUs) revolutionize Deep Learning

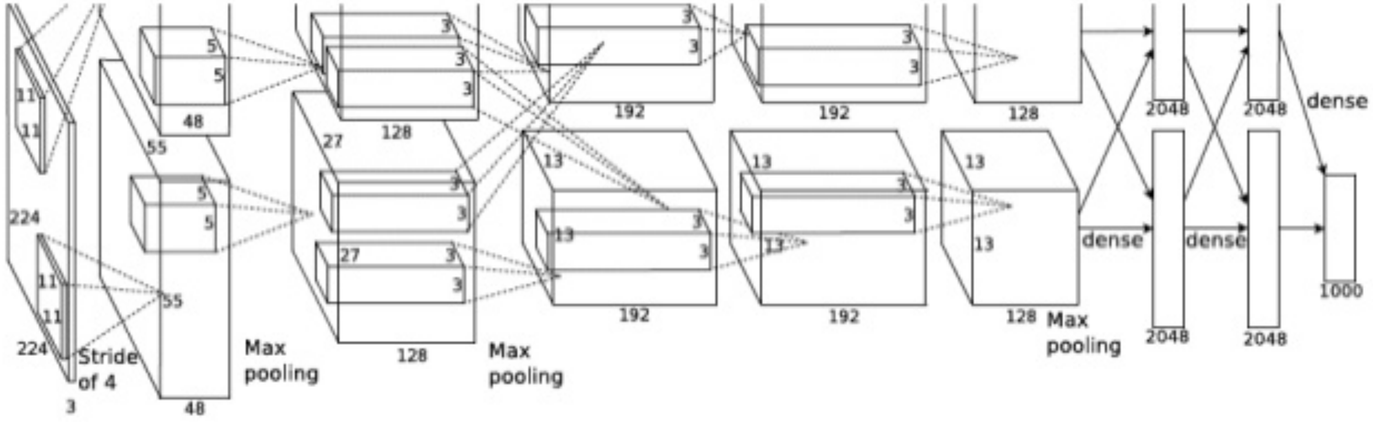
- GPUs first introduced in 2006 for DL
- Order of magnitude increase in speed of training
- Nvidia is the major player; Intel and others lagging behind.
- New Tensor Processing Units (TPUs) being offered by Google

ImageNet Dataset (2011): The Largest Hand-Labeled Dataset in the World

Convnets dominate ImageNet Challenge (2012)

AlexNet

- Similar framework to LeCun'98 but:
  - Bigger model (7 hidden layers, 650,000 units, 60,000,000 params)
  - More data (10^6 vs. 10^3 images)
  - GPU implementation (50x speedup over CPU)
    - Trained on two GPUs for a week

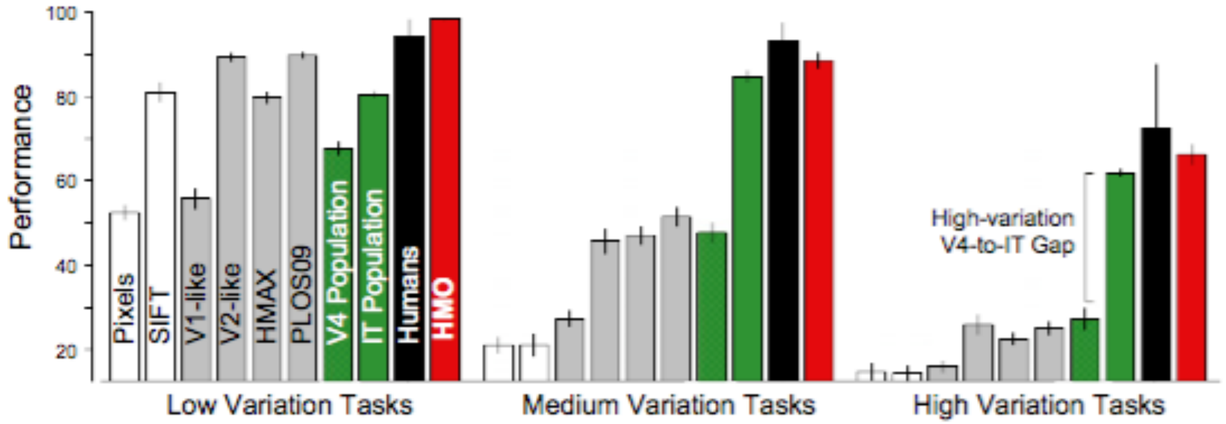


A. Krizhevsky, I. Sutskever, and G. Hinton,
ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012

Deep Learning:
Recent Applications to Neuroscience

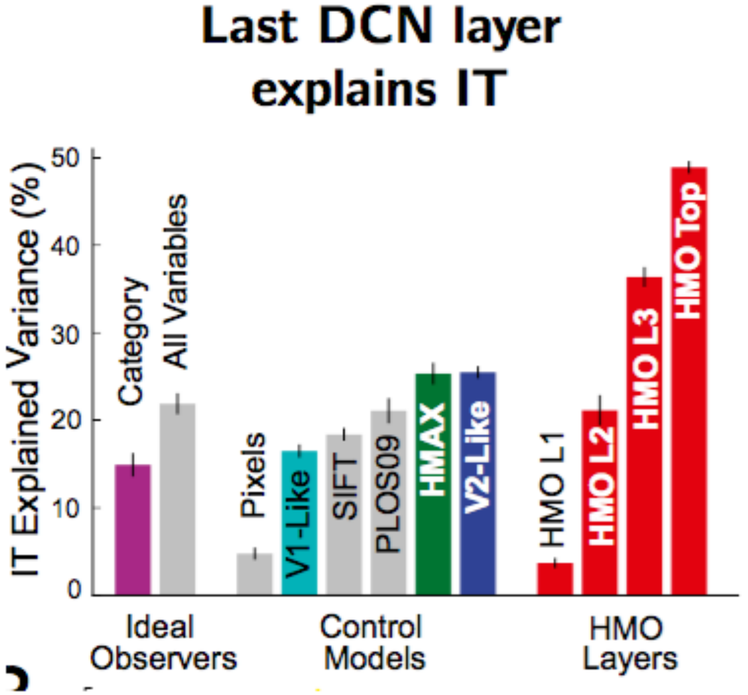
Recent Finding: Best model of neural activations in ventral stream (V1 to IT) is a DCN

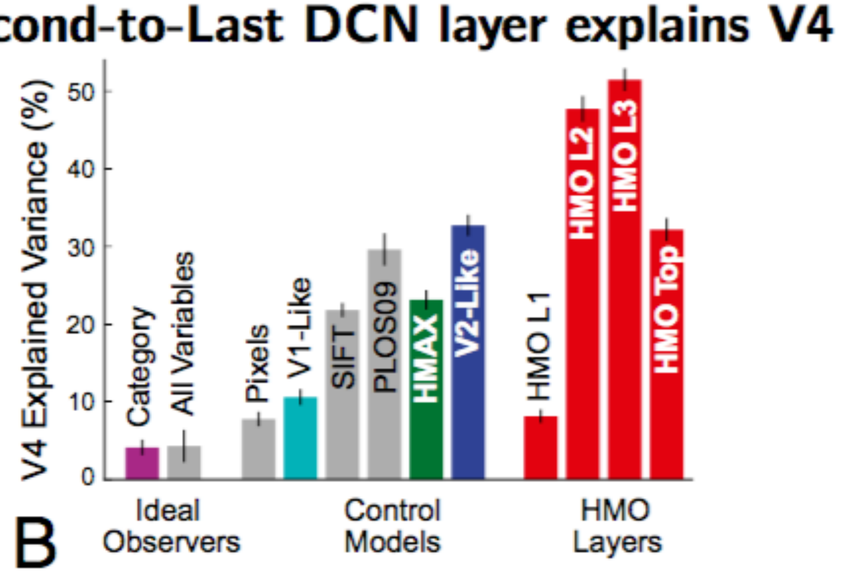
- Low nuisance variation: all models perform well
- High nuisance variation: IT and DCN perform best by far



D. Yamins, H. Hong, C. Cadieu, E. Solomon, D. Seibert, and J. J. DiCarlo.
Performance-optimized hierarchical models predict neural responses in higher visual cortex. PNAS 2014

Deep Learning:
Recent Applications to Neuroscience





D. Yamins, H. Hong, C. Cadieu, E. Solomon, D. Seibert, and J. J. DiCarlo.
Performance-optimized hierarchical models predict neural responses in higher visual cortex. PNAS 2014

The Implication

A better understanding of Deep Convolutional Nets may lead to
a better understanding of the Visual Cortex

Conclusions

- History of NNs touches upon many different fields and ideas:
  - Neuroscience, Cognitive Science, Mind-Body problem
  - Boolean functions, logic, expressive power
  - “Machine” Learning, Optimization
- Chock full of interesting ideas that were *far ahead of their time:*
  - Many of them are resurging now —> Final Projects or your own research?