CISC452/CMPE452/COGS400/CISC874
ANN – Why?

Farhana Zulkernine

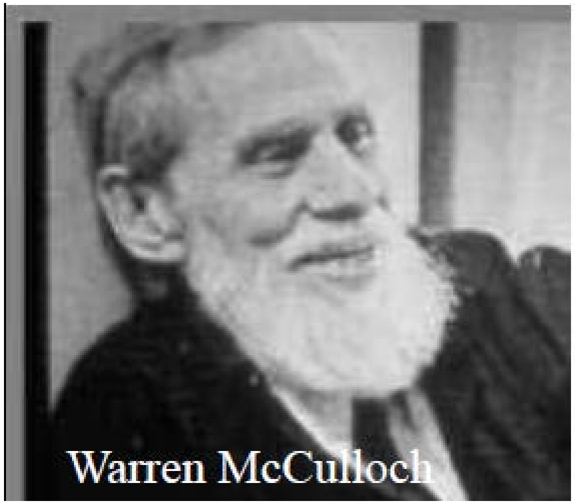
Why Cognitive Modeling?

1. To understand cognitive processes better.
   - Understanding language, learning, perception, recognition, memory, logic processing, computation, attention, decision making
2. To computationally implement a cognitive process.
   - Create a model of the mind.
     - Can we build an actual mind some day?
3. Compare and evaluate the various explanations of the cognitive processes.
4. Predict outcomes of cognitive processes.
5. May not be fully accurate to account for human errors and uncertainty – different from computational modeling.

Earlier Approaches

- Rule based approaches
  - Need an expert to understand and define rules
  - Uses symbols (physical symbol systems – PSS) and has serial execution
- Constraints
  - Multiple if-then-else coding can get very complicated
  - Difficult to extract rules – tacit knowledge which is difficult to transfer
  - Cannot implement asynchronous data flow and parallel processing

The Alternative Approach…



- Warren McCulloch (1899-1969) was an army psychiatrist who turned into neuropsychologist
- Assembled research group with Walter Pitts, Jerry Lettvin
- Pursued cybernetic model of the brain at MIT – proposed the first ANN model

Learning in Brain Neurons

- Influenced by survival, emotion, and memory of past experience (good and bad)
- Influenced by the demand of physical system
- Influenced by evolution (rote learning, copy others)
- Layout of neurons and synaptic connections change accordingly
  - Repeated simultaneous firing of neurons strengthens synapses
    - Skill learning by doing something repeatedly
    - Skill learning, rewarding experience
  - Chemicals within the brain changes to generate stimuli for firing of neurons

Learning in ANN

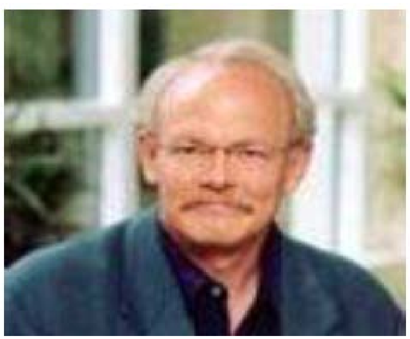
- Reorganize or restructure ANN
- Change firing threshold
- Increase connection strength by
  - increasing associated weight
- ANN has 3 main types of learning
  - Error correction learning
    - Learn from mistakes
  - Reinforcement/Correlation/Hebbian learning
    - Learn if rewarded – Named after Donald Heb
  - Competitive learning
    - Variation of Hebbian learning

The First Famous ANN

- ANNs implement Parallel Distributed Processing (PDP) where processing is done in a distributed network in parallel.
- The first ANN to achieve prominence was, however, hardwired in advance.
- This is the Interactive Activation Model, presented by McClelland and Rumelhart (1980, 1981).

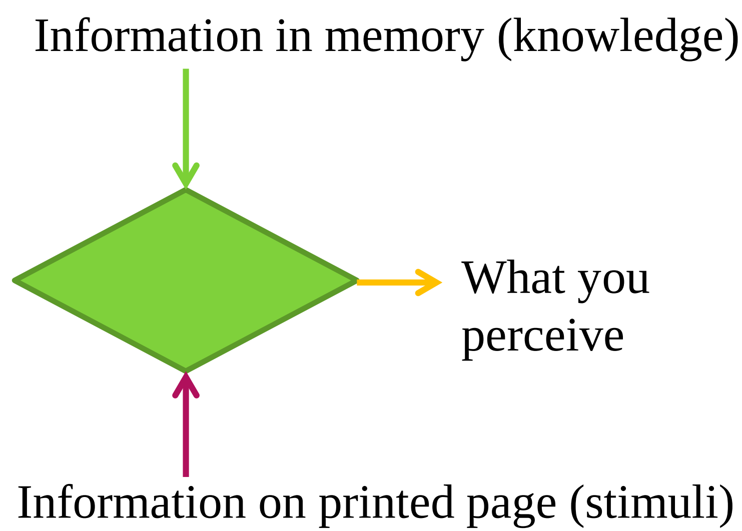
Parallel Distributed Processing (PDP)

- PDP also explains how *top-down and bottom-up information is combined in a perceptual task, like word recognition.*



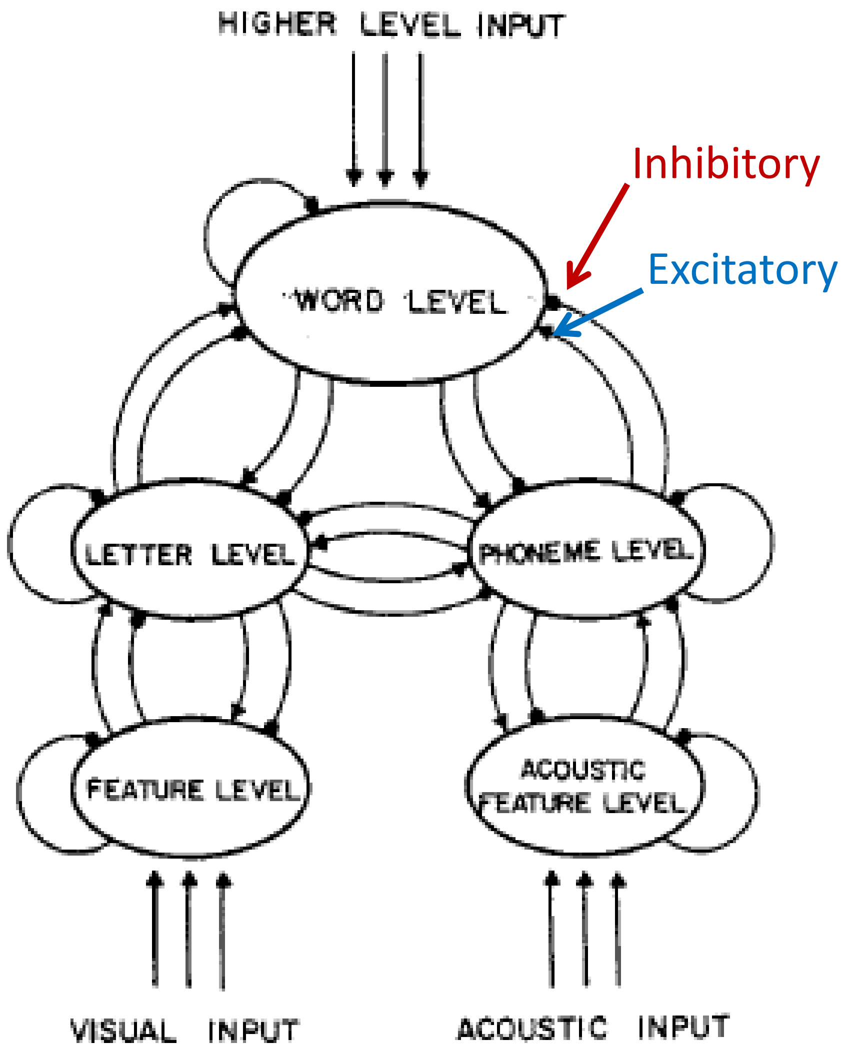
Jim McClelland

Dave Rumelhart



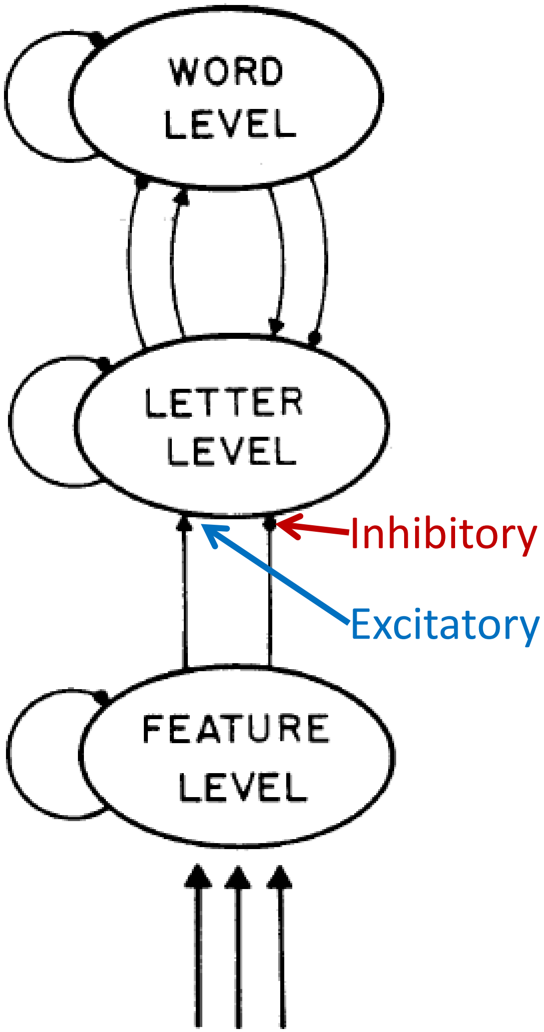
Word Perception

- Some of the processing levels involved in visual and auditory word perception has interconnection among nodes in the same layer (McClelland and Rumelhart, 1981).
- Assumptions: 3 levels where top and bottom level gets input from higher level and visual and acoustic inputs accordingly.



IAM – Simple Model – Part 1

- To make it simple and reduce interactions, initial model (part 1) considers
  - Reciprocity of activation between word and letter in paper on part 1.
  - Ignores phonological processes.
- Features are given as inputs.
- Discrete rather than continuous time.
- A word and letter level node connects with neighbouring word or letter nodes at the same or adjacent levels.



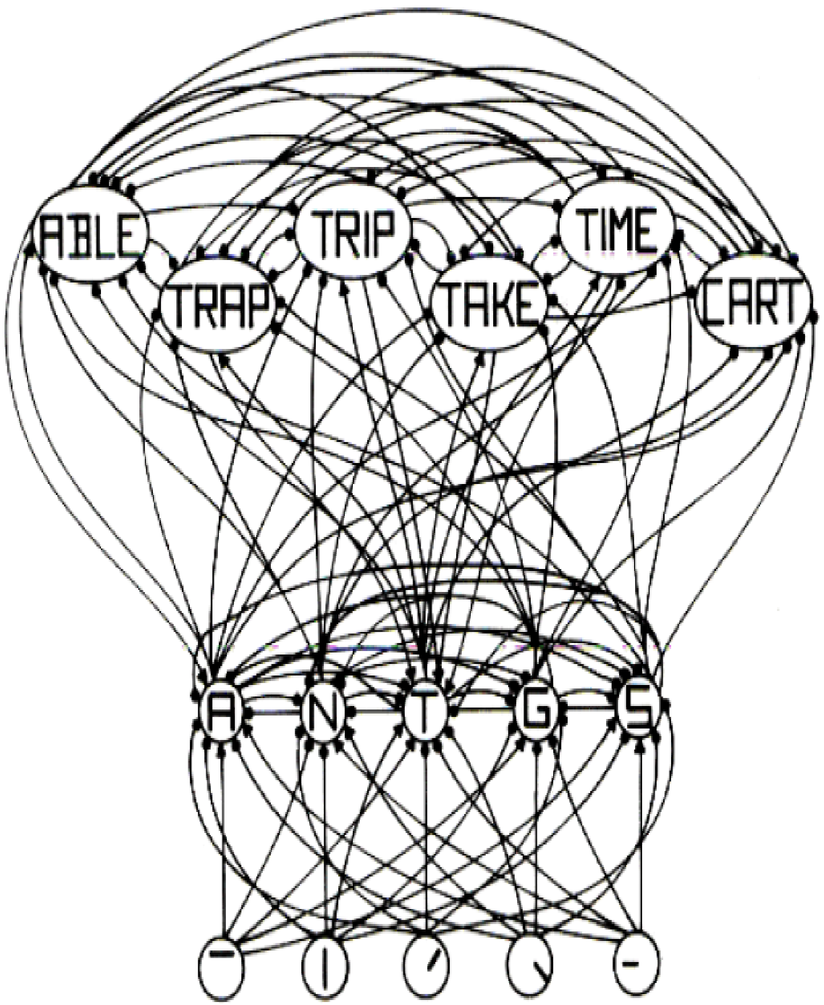
Input Features for Letters

- Simplified feature analysis of input font.
- Limited lexicon.
- Input consists of the visual feature detectors that are on for each letter.

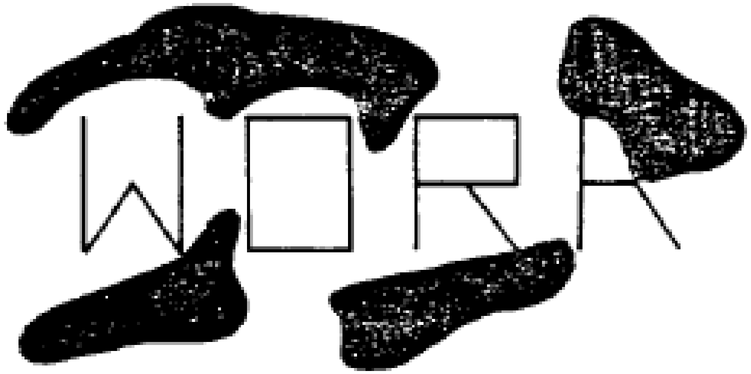
Font and feature analysis process is from Rumelhart 1970 and from Rumelhart and Siple 1974.

IAM Model – Example for ‘T’

- Activation is a positive real value at time *t*, given by $a_i(t)$
- In absence of activation, a previously activated node decays back at a rate θ_i to inactive state with a resting state $a <= 0$.
- Nodes for high frequency words have resting levels higher than low frequency words but varies by r_i.

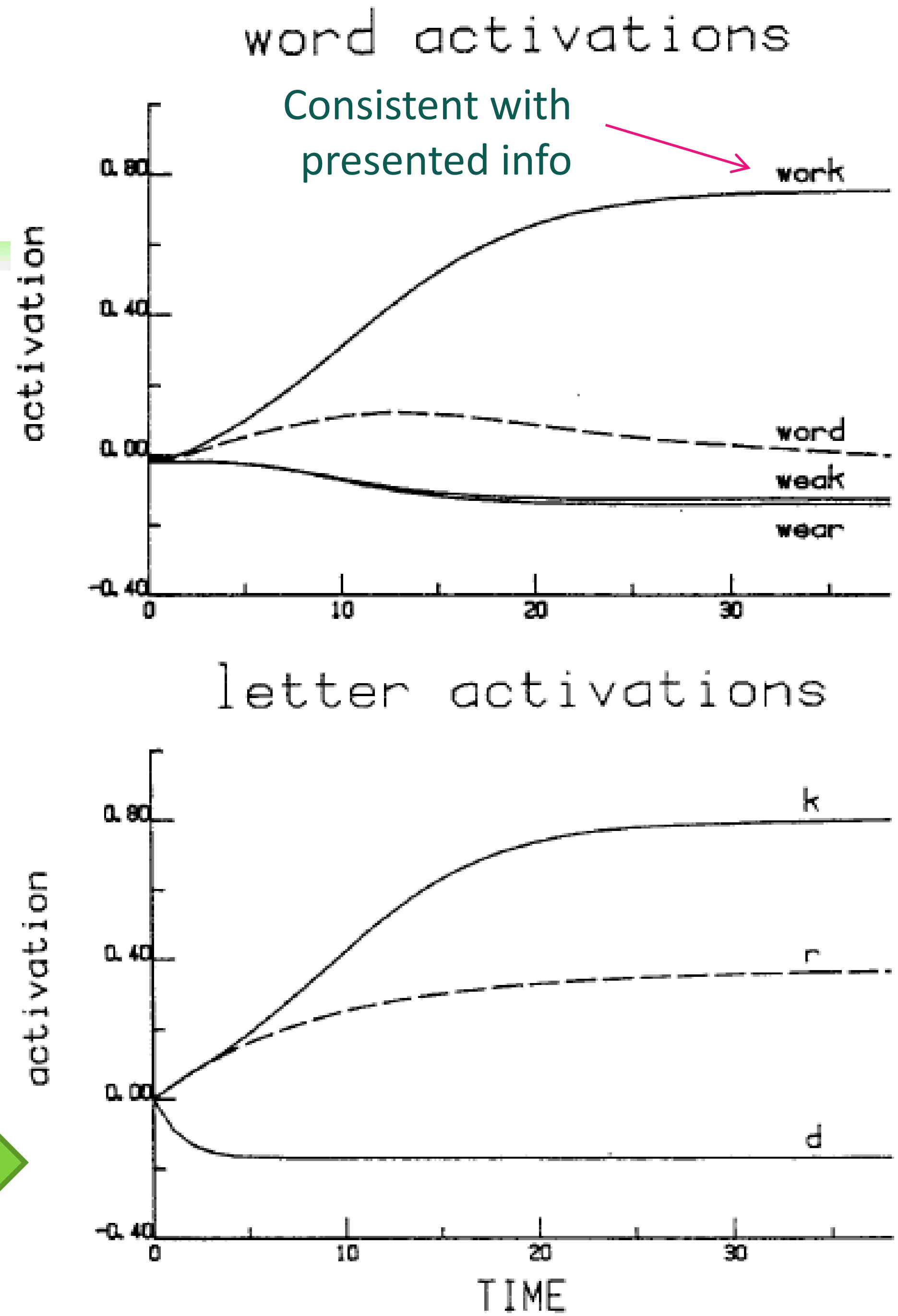


Sample Run of the Simulation Model



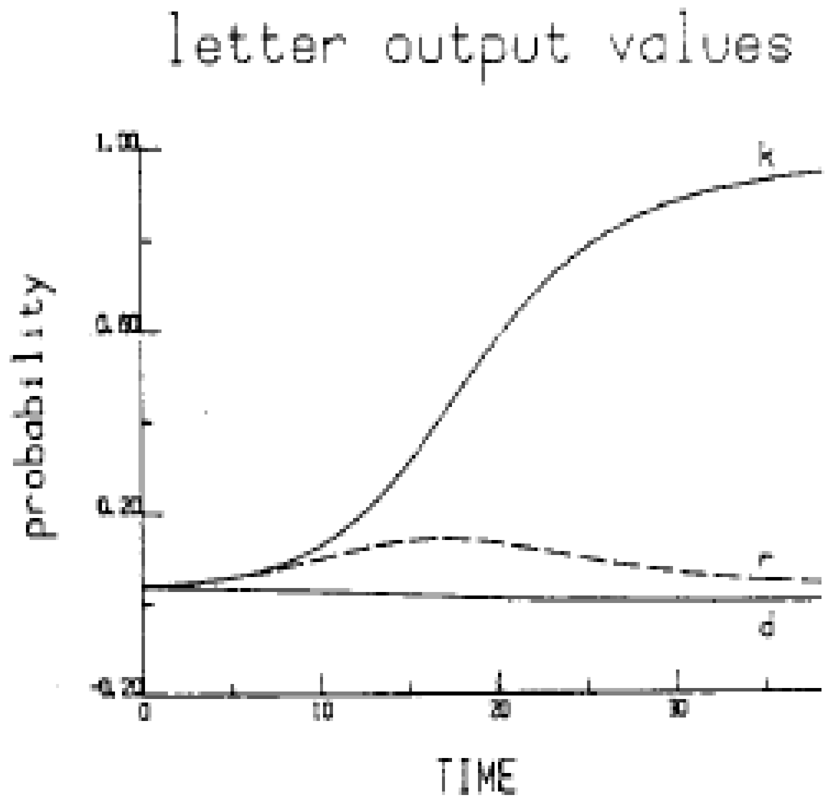
- W, O, R extracted but the last letter can be R or K.

Graph shows time-course of activations for selected nodes at word and letter levels respectively.



Output Probability

- In the previous simulation, word-to-letter and letter-to-letter level inhibitions are set to 0 which is why k (feedback from both levels) and r (from only bottom) both have moderately high values.
- In this one inhibitory signals are active.



Final output as probability of the different letters – wait until becomes stable

IAM – Context Effects in Letter Perception

- IAM is a model of context effects in perception of letters.
- Perception results from excitatory and inhibitory interactions of detectors of visual features, letters, and words.

1. Visual input excites detectors for visual features in the display
2. Active features *inhibit other feat*ures and
   - Activate those letters which contain the features and
   - Inhibit letters which do not contain the features.
3. Active letters *inhibit other letters* and
   - Activate words which contain the letters and
   - Inhibit words which do not contain the letters.

Feedback Cycles in the Model

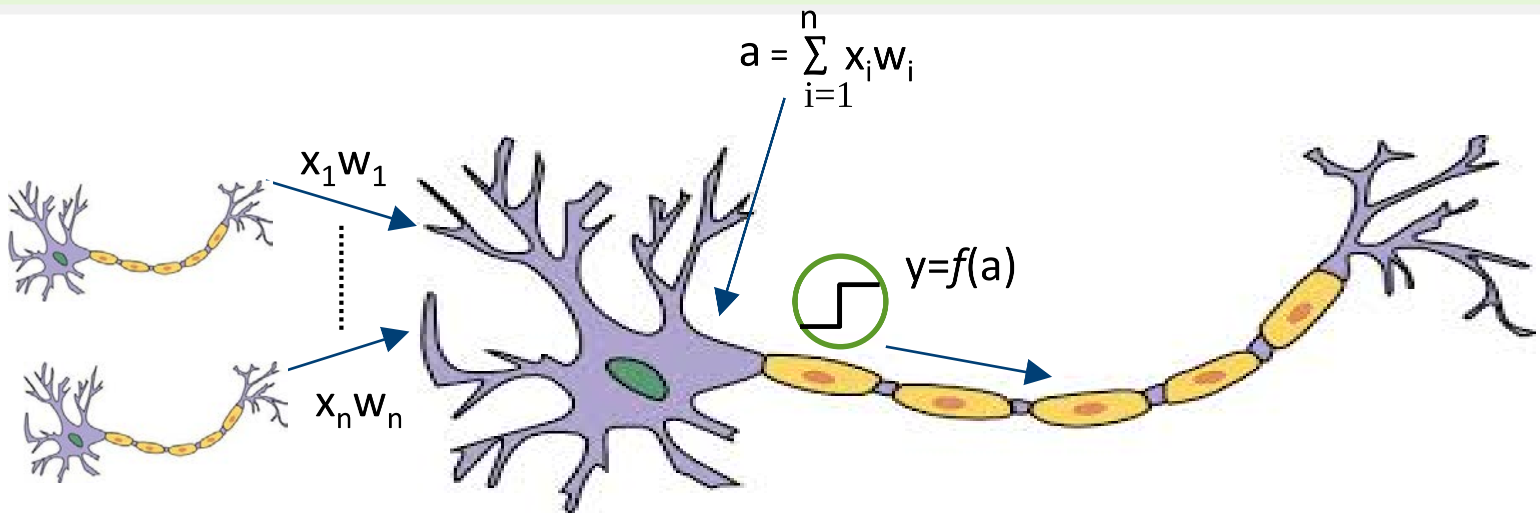
4. Active word detectors mutually *inhibit* each other then
   - Send feedback to letter level to activate the letters they contain and inhibit letters they do not contain.
   - Feedback strengthens perception of constituent letters.
5. Active letter detectors
   - Send feedback to feature level to activate the features they contain and inhibit features they do not contain.
   - Feedback strengthens perception of constituent features.
6. And thus it cycles between the feature, letter and word…

- Letters in words are more perceptible because they get the reinforcement activation from word level.

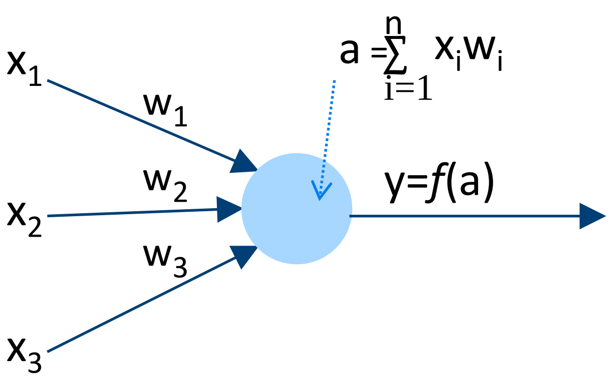
ANN Design

Structure, firing and learning in ANN

Learning in the Neuron



McCulloch and Pitts Neuron Model



The weights w_i take on *real values* $w_i \in \mathbb{R}$

Activation is the weighted sum of all incoming potentials.

f(a) can be any function that generates a spike (high value) at a given threshold value θ to mimic the scenario of *Action Potential*.

The Activation Function

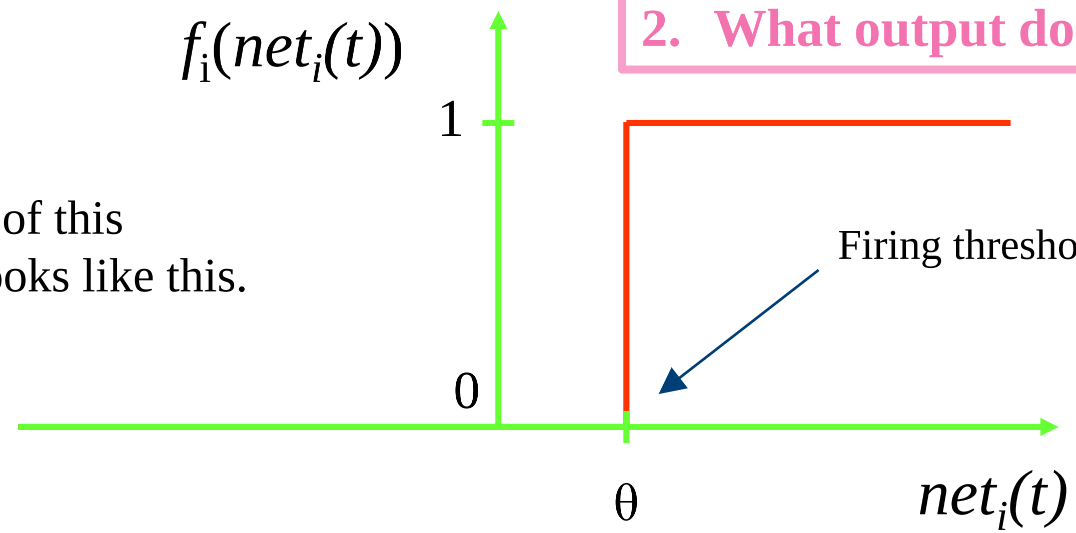
One possible choice is a threshold function:

Therefore, we call this a threshold neuron.

$$f_i(\text{net}_i(t)) = 1, \quad \text{if} \quad \text{net}_i(t) \geq \theta$$
$$= 0, \quad \text{otherwise}$$

1. **When does it fire?**
2. **What output does it generate?**

The graph of this function looks like this.



Output Functions $f(net_i)$

- The simplest node functions are:

1. **Identity**, f(net) = net, and its non-negative variant f(net) = max(0, net)

2. Constant functions f(net) = c

3. Signum function

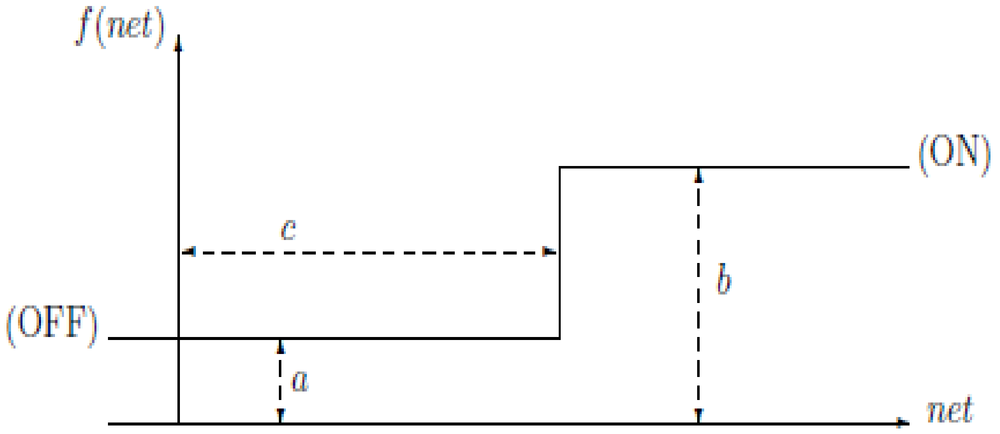
$$f(net) = \begin{cases} +1 & \text{if } net > 0 \\ -1 & \text{if } net < 0 \\ 0 & \text{if } net = 0 \end{cases}$$

4. Step Function

- Simplest function that captures the idea of a "firing threshold"
- Can be used as a class identifier
- Problem: Very small change in $net_i(t)$ can cause a spike and hence change the output

$f(\text{net}) = a$ if $net < c$

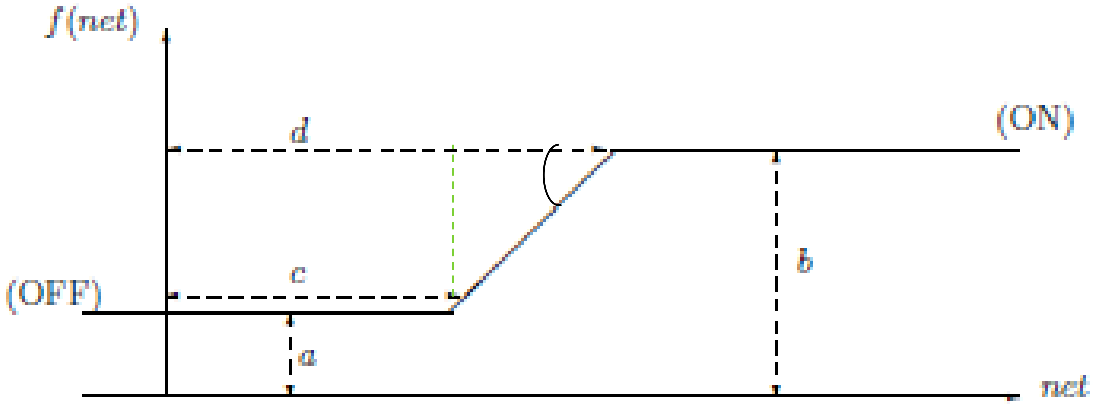
$= b$ if $net >= c$



5. a) Ramp Function

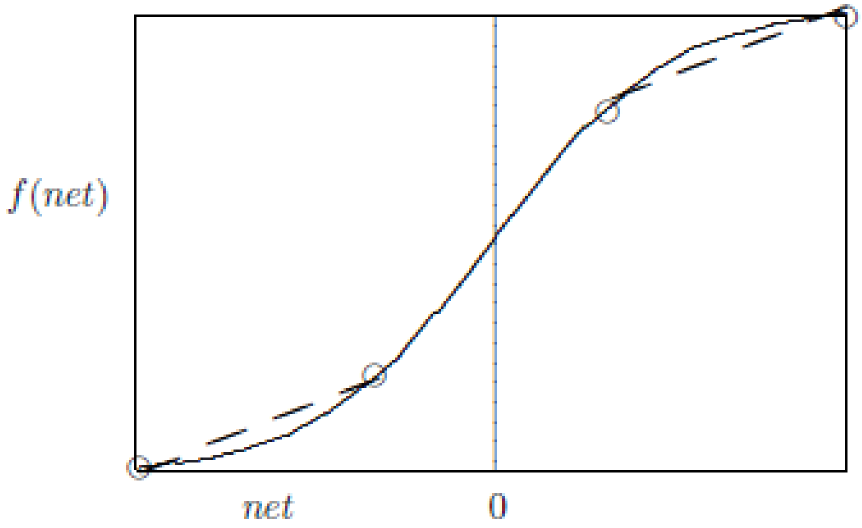
- The ramp function is continuous and almost everywhere differentiable in exchange of the simple ON/OFF description of the output.

$$f(net) = \begin{cases} a & \text{if } net \leq c \\ b & \text{if } net \geq d \\ a + \frac{(net-c)(b-a)}{d-c} & \text{otherwise} \end{cases}$$



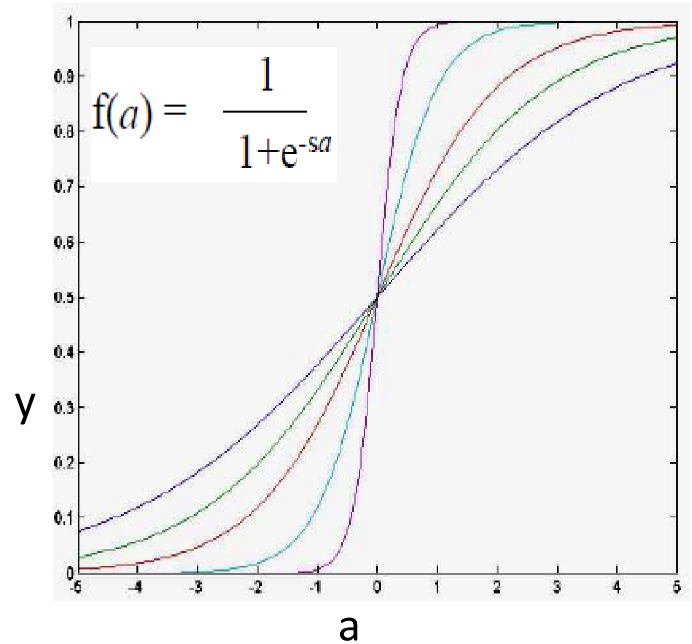
5. b) Piecewise Linear Functions

- Consist of finite number of linear segments, and are thus differentiable almost everywhere.
- Easier to compute than general nonlinear functions such as sigmoid functions.
- Can be used to avoid sudden change in output like the step function (from 0 to 1).



6. Sigmoid Function

- These functions are continuous and differentiable everywhere, and asymptotically approach saturation values (0 and 1 as shown in the picture)
- The parameter *s* controls the slope of the sigmoid function. Greater *s* value will give steeper curve.



$$\lim_{net \to +\infty} f(net) = 1$$

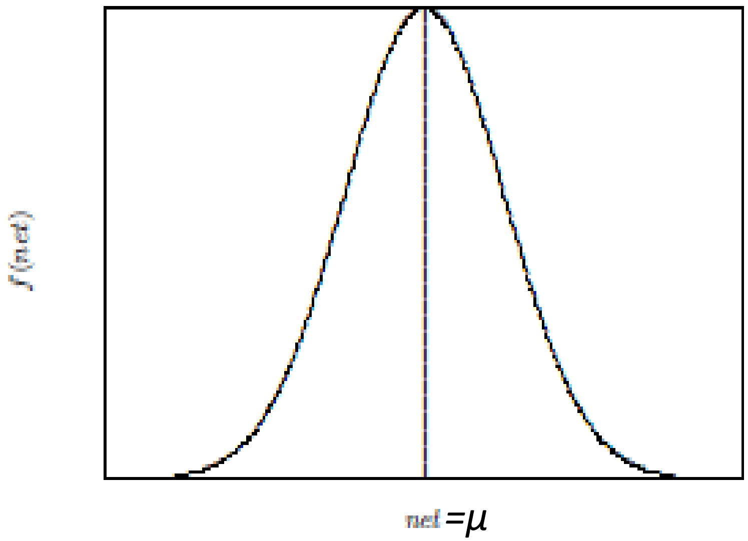
$$\lim_{net \to -\infty} f(net) = 0$$

7. Gaussian Functions

- Continuous bell-shaped functions.
- Also called 'radial-basis' function.
- *f*(net) asymptotically approaches 0 (or some constant) for large magnitudes of *net*, with a single maximum for

  $net = \mu$, say $\mu = 0$.

  Greater σ → wider curve.



$$f(net) = \frac{1}{\sqrt{2\pi}\sigma} exp[-\frac{1}{2}(\frac{net - \mu}{\sigma})^2]$$

Summary

- First models of ANN didn’t support learning
- Use different threshold and output function based on application requirements
- Differentiability of output functions is desired to manipulate behavior by adjusting parameter values