
# Neural Networks

## 1. Introduction

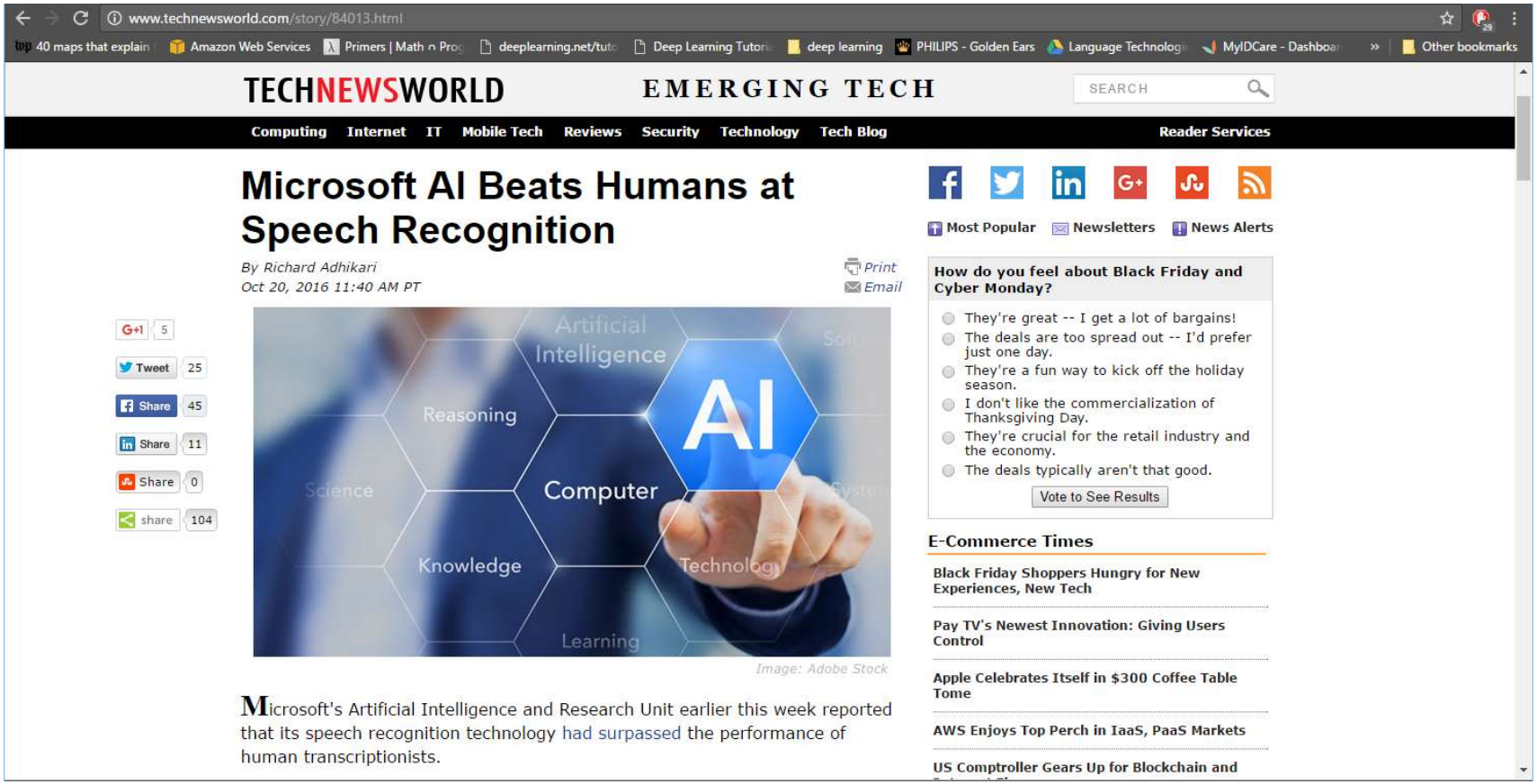
## Fall 2018

# Neural Networks are taking over!

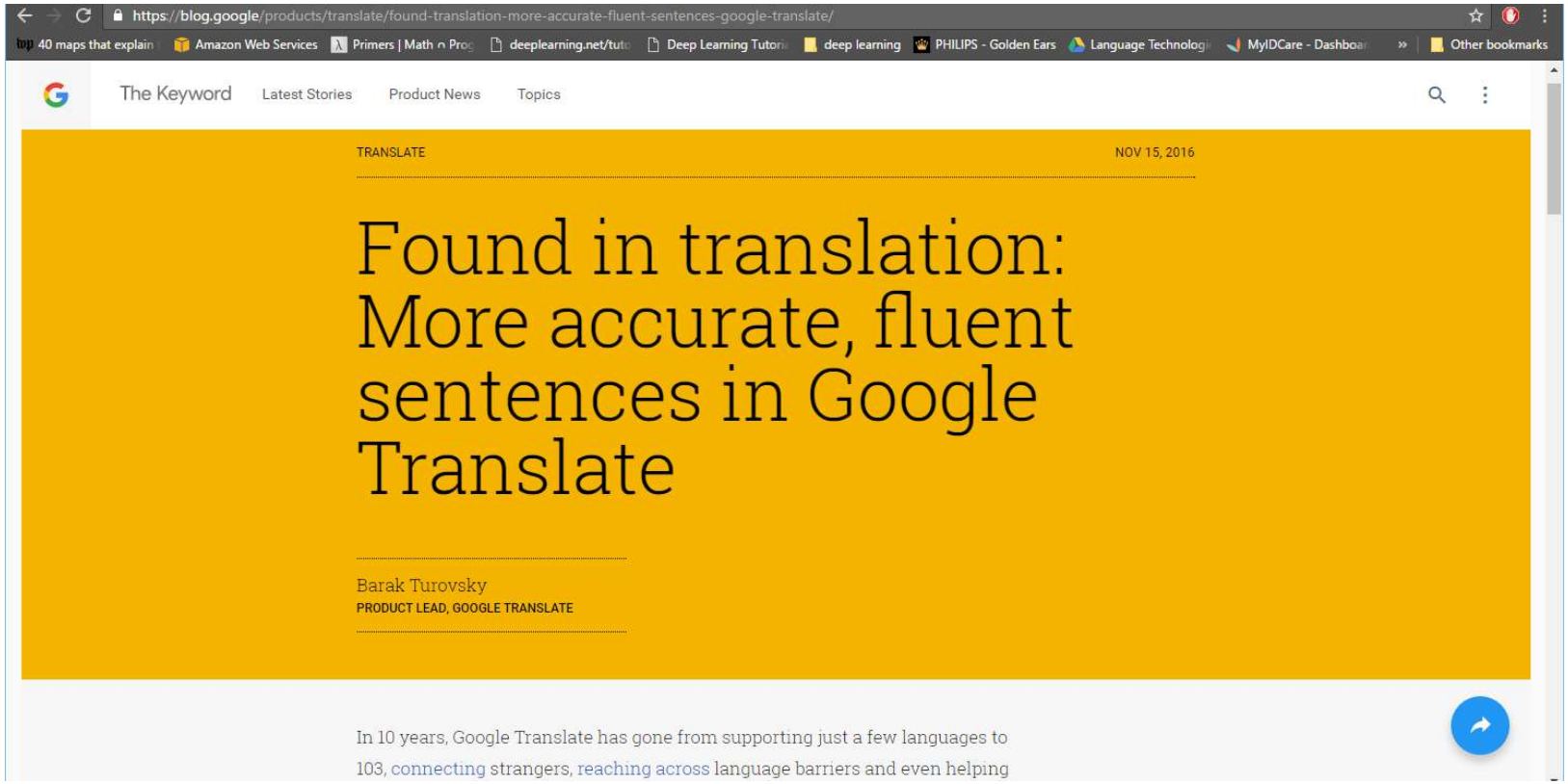
- Neural networks have become one of the major thrust areas recently in various pattern recognition, prediction, and analysis problems

- In many problems they have established the state of the art
  - Often exceeding previous benchmarks by large margins

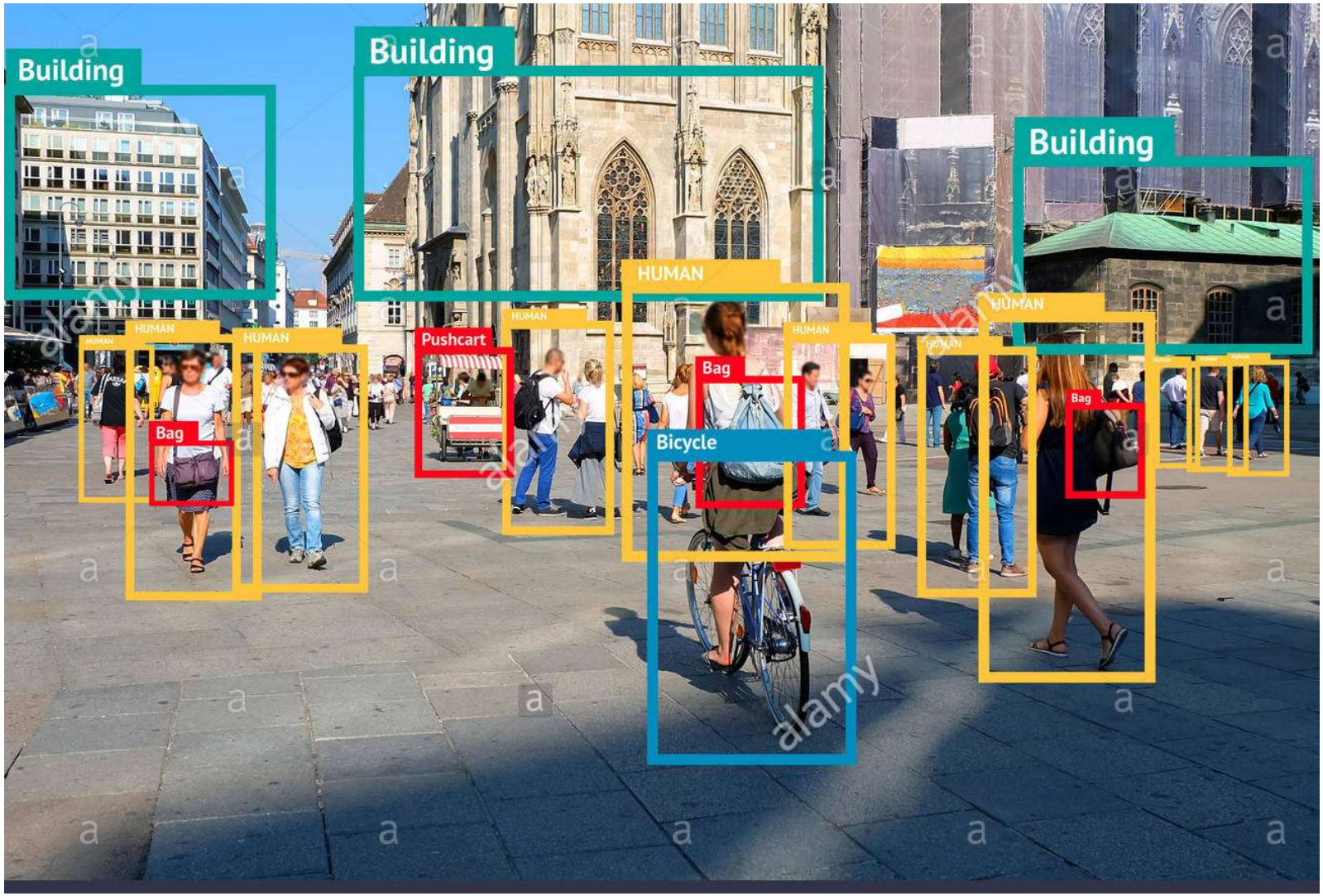
# Recent success with neural networks



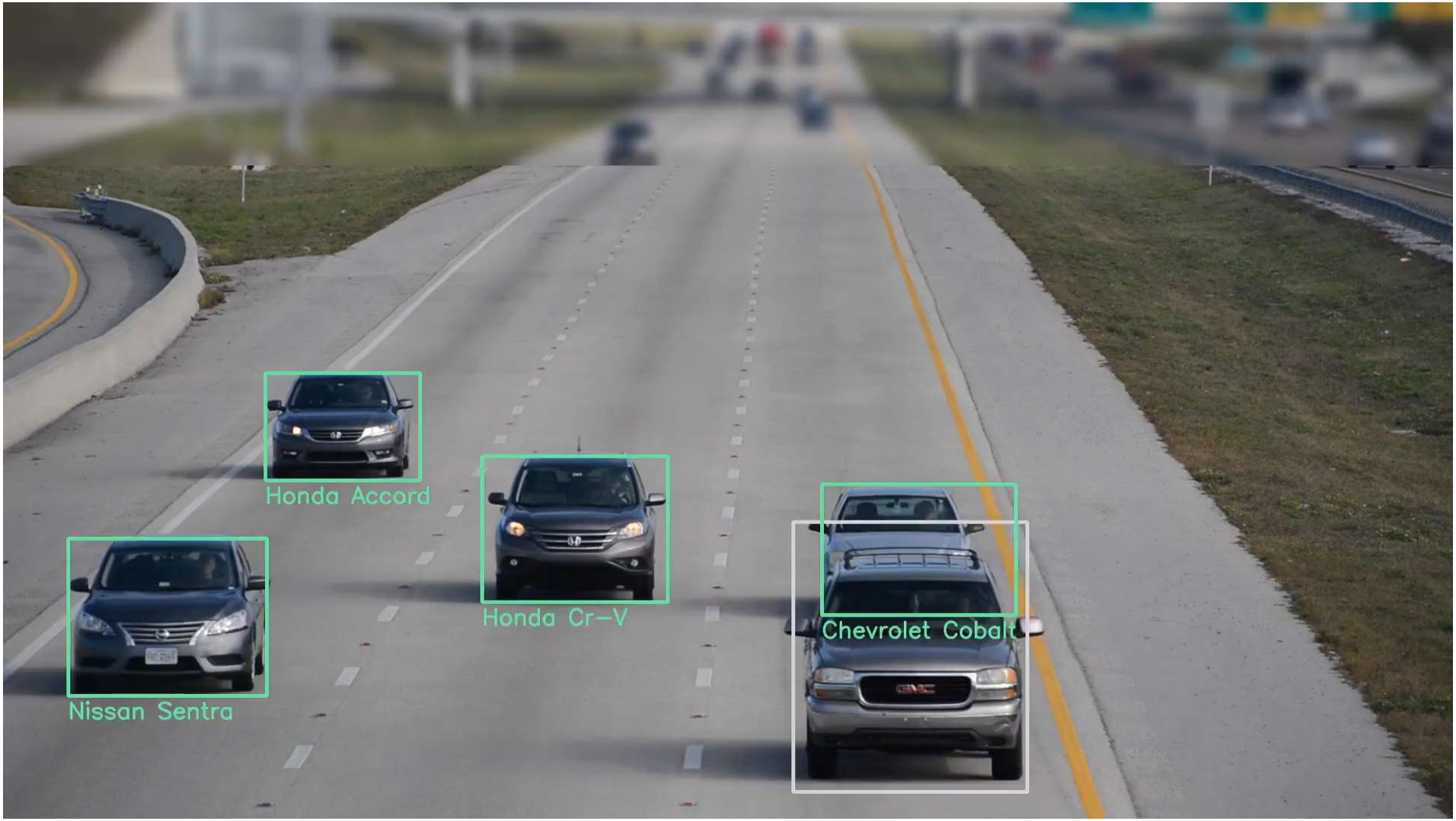
# Recent success with neural networks



# Image segmentation and recognition

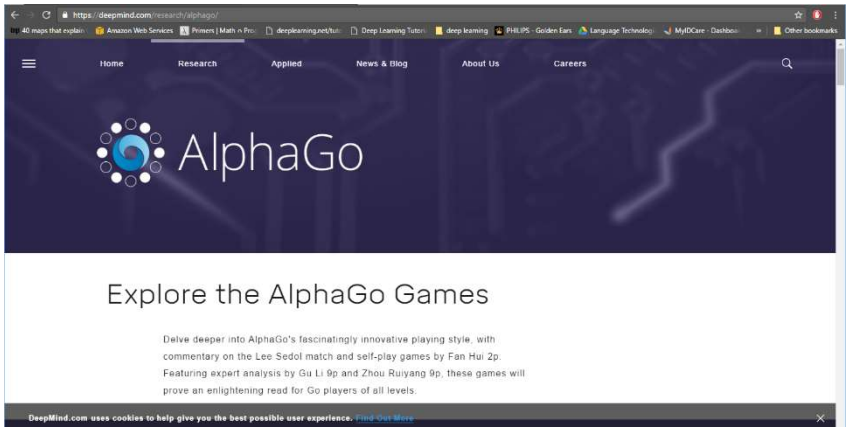


# Image recognition




https://www.sighthound.com/technology/

# Recent success with neural networks



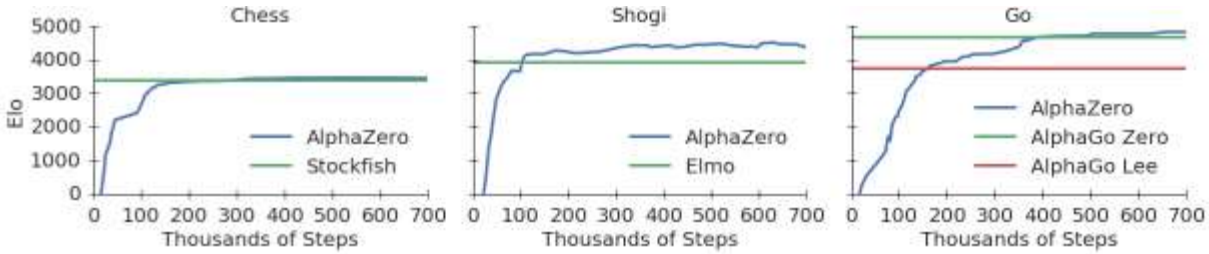
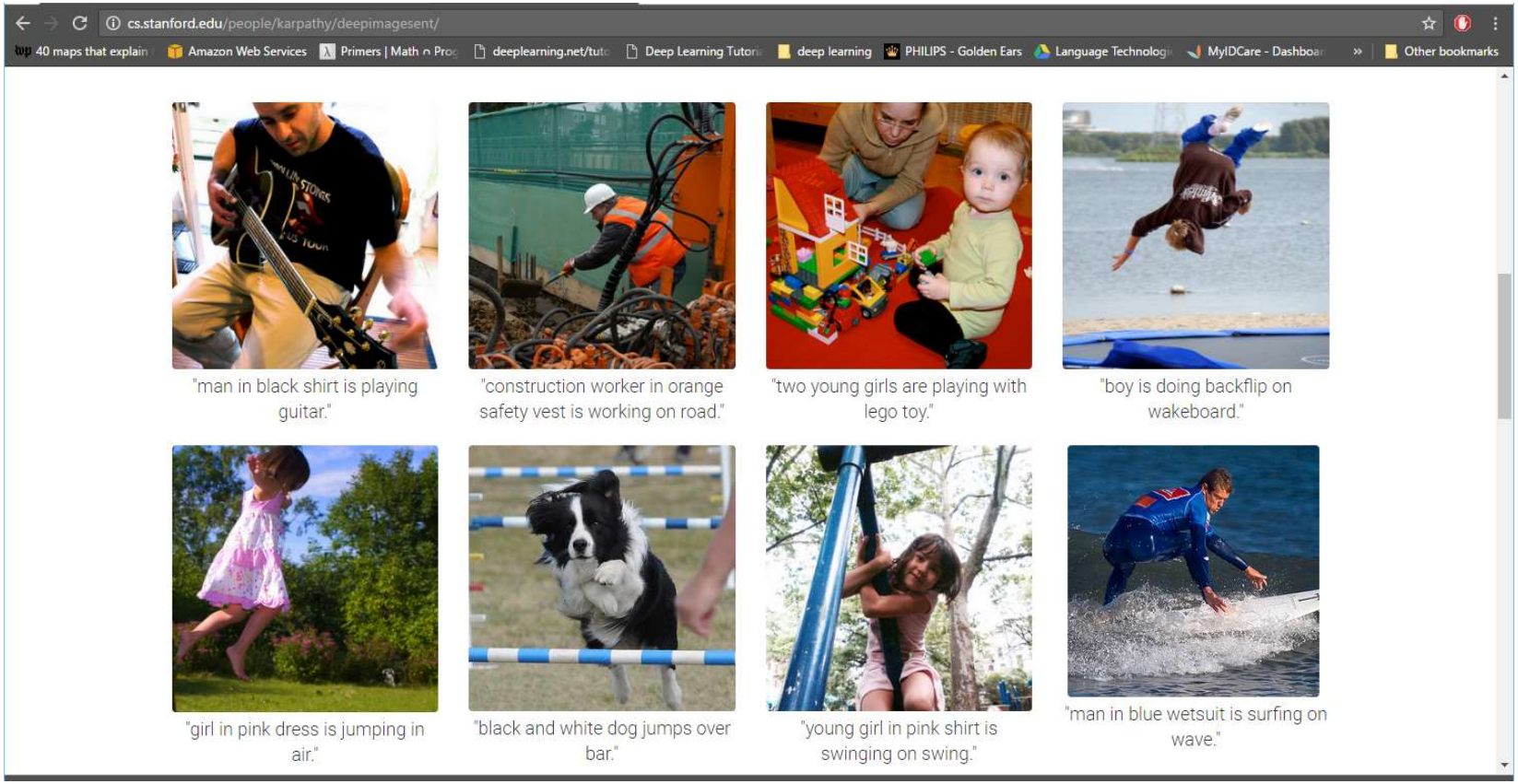




Figure 1: Training *AlphaZero* for 700,000 steps. Elo ratings were computed from evaluation games between different players when given one second per move. **a** Performance of *AlphaZero* in chess, compared to 2016 TCEC world-champion program *Stockfish*. **b** Performance of *AlphaZero* in shogi, compared to 2017 CSA world-champion program *Elmo*. **c** Performance of *AlphaZero* in Go, compared to *AlphaGo Lee* and *AlphaGo Zero* (20 block / 3 day) (*29*).

# Success with neural networks




- Captions generated entirely by a neural network

# Successes with neural networks

- And a variety of other problems:
  - From art to astronomy to healthcare..
  - and even predicting stock markets!

# Neural nets can do anything!

# Neural nets and the employment market


This guy didn't know about neural networks (a.k.a deep learning)

This guy learned about neural networks (a.k.a deep learning)

# Objectives of this course

- Understanding neural networks
- Comprehending the models that do the previously mentioned tasks
  - And maybe build them
- Familiarity with some of the terminology
  - What are these:
    - http://www.datasciencecentral.com/profiles/blogs/concise-visual-summary-of-deep-learning-architectures
- Fearlessly design, build and train networks for various tasks

- *You will not become an expert in one course*

# Course learning objectives: Broad level

- Concepts
  - Some historical perspective
  - Types of neural networks and underlying ideas
  - Learning in neural networks
    - Training, concepts, practical issues
  - Architectures and applications
  - Will try to maintain balance between squiggles and concepts (concept >> squiggle)
- Practical
  - Familiarity with training
  - Implement various neural network architectures
  - Implement state-of-art solutions for some problems
- Overall: Set you up for further research/work in your research area

# Course learning objectives: Topics

- Basic network formalisms:
  - MLPs
  - Convolutional networks
  - Recurrent networks
  - Boltzmann machines
- Some advanced formalisms
  - Generative models: VAEs
  - Adversarial models: GANs
- Topics we will touch upon:
  - Computer vision: recognizing images
  - Text processing: modelling and generating language
  - Machine translation: Sequence to sequence modelling
  - Modelling distributions and generating data
  - Reinforcement learning and games
  - Speech recognition

# Reading

- List of books on course webpage
- Additional reading material also on course pages

# Instructors and TAs

- Instructor: Me
  - bhiksha@cs.cmu.edu
  - x8-9826
- TAs:
  - List of TAs, with email ids on course page
  - We have TAs for the
    - Pitt Campus
    - Kigali,
    - SV campus,
    - Doha campus
  - Please approach your local TA first
- Office hours: On webpage
- http://deeplearning.cs.cmu.edu/

# Logistics

- Most relevant info on website
  - Including schedule
- Short video with course logistics up on youtube
  - Link to be posted shortly on Piazza
  - Please watch: Quiz includes questions on logistics
- Repeating some of it here..

# Logistics: Lectures..

- Have in-class and online sections
  - Including online sections in Kigali, SV and Doha
- Lectures are being streamed
- Recordings will also be put up and links posted
- Important that you view the lectures
  - Even if you think you know the topic
  - Your marks depend on viewing lectures

# Lecture Schedule

- On website
  - The schedule for the latter half of the semester may vary a bit
    - Guest lecturer schedules are fuzzy..
- Guest lectures:
  - TBD
    - Scott Fahlman, Graham Neubig, Richard Stern, Joseph Keshet

# Recitations

- We will have 13 recitations
- Will cover implementation details and basic exercises
  - Very important if you wish to get the maximum out of the course
- Topic list on the course schedule

- *Strongly recommend attending all recitations*
  - *Even if you think you know everything*

# Recitations Schedule

- 27 August: AWS
- 7 Sep
- 14 Sep
- 21 Sep
- 28 Sep
- 5 Oct
- 12 Oct
- 19 Oct
- 26 Oct
- 2 Nov
- 9 Nov
- 16 Nov
- 23 Nov
- 30 Nov
- 7 Dec

# Grading

- Weekly multiple-choice Quizzes (24%)
- 5 homeworks (41%)
  - Preparatory homework (HW0 – 1%)
  - Basic MLPs
  - CNNs
  - RNNs
  - Sequence to sequence modelling
- One project (35%, for 11-785, does not apply to 11-485)
  - 5% Proposal
  - 5% Mid-term report
  - 15% Project presentation
  - 10% Final report

# Weekly Quizzes

- Weekly quizzes
  - 10 multiple-choice questions
  - Relate to topics covered that week
  - Released Friday, closed Saturday night
    - This may occasionally shift
- There will be 14 total quizzes
  - We will consider the best 12
  - This is expected to account for any circumstance-based inability to work on quizzes
    - You could skip up to 2

# Lectures and Quizzes

- Slides often contain a lot more information than is presented in class

- *Quizzes will contain questions from topics that are on the slides, but not presented in class*

- *Will also include topics covered in class, but not on online slides!*

# Homeworks

- Homeworks come in two flavors
  - Autograded homeworks with deterministic solutions
    - You must upload them to autolab
  - Kaggle problems
    - You compete with your classmates on a leaderboard
    - **We post performance cutoffs for A, B and C**
      - **If you achieved the posted performance for, say "B", you will at least get a B**
      - **A+ == 105 points (bonus)**
      - **A = 100**
      - **B = 80**
      - **C = 60**
      - **D = 40**
      - **No submission: 0**
    - **Actual scores are linearly interpolated between grade cutoffs**
      - Interpolation curves will depend on distribution of scores

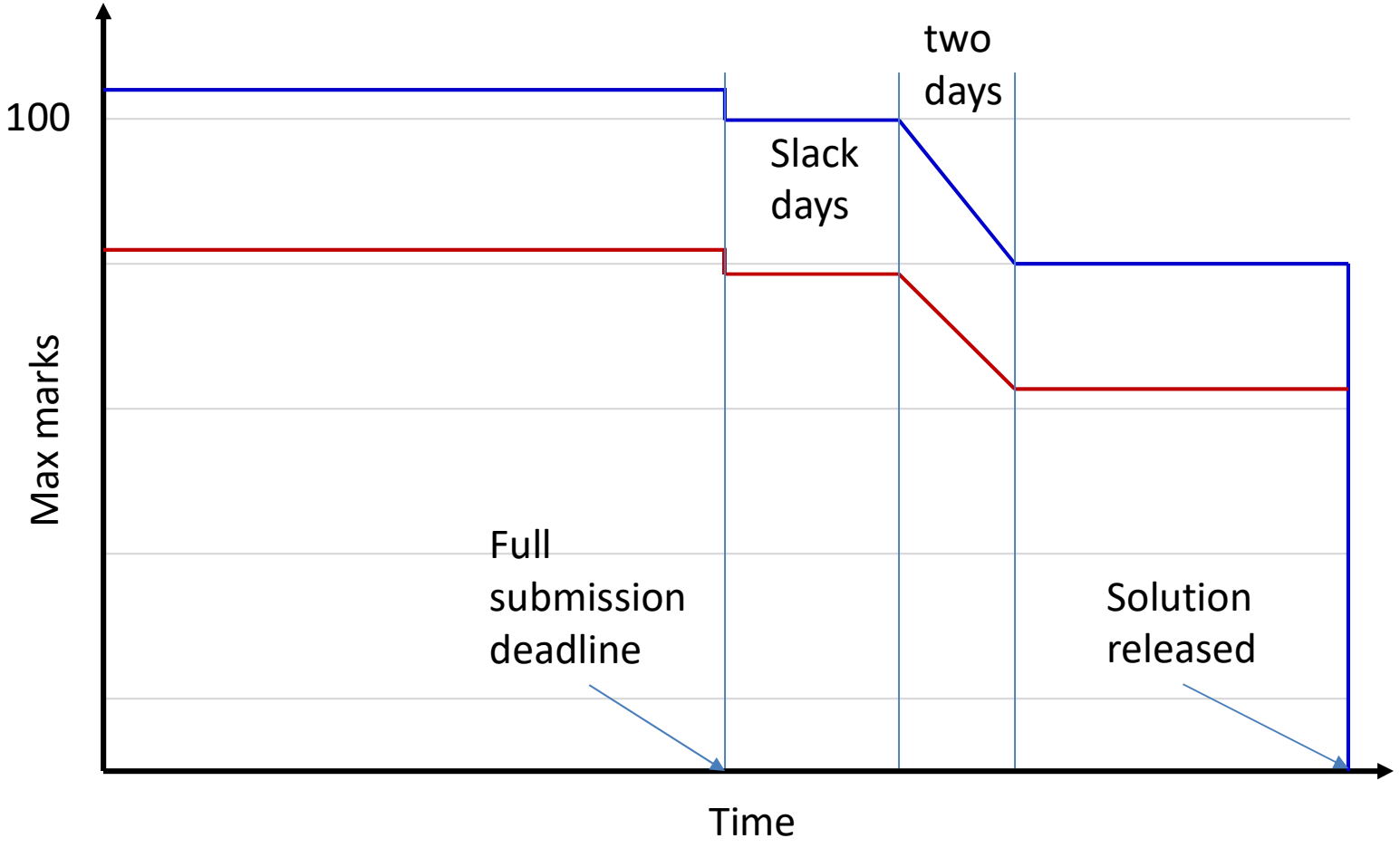
# Homework Deadlines

- Multiple deadlines

- Separate deadline for Autograded deterministic component
- Kaggle component has multiple deadlines
  - *Initial submission* deadline : If you don't make this, all subsequent scores are multiplied by 0.8
  - *Full submission deadline:* Your final submission must occur before this deadline to be eligible for full marks
  - *Drop-dead deadline:* Must submit by here to be eligible for *any* marks
    - Day on which solution is released

- Homeworks: Late policy
  - Everyone gets up to 5 total slack days (does not apply to initial submission)
  - You can distribute them as you want across your HWs
    - You become ineligible for "A+" bonus if you're using your grace days for Kaggle
  - Once you use up your slack days, you lose 10% of your points for 1 day of delay, 20% for two, and get no points beyond the drop-dead deadline
  - Kaggle: Kaggle leaderboards stop showing updates on full-submission deadline
    - But will continue to privately accept submissions until drop-dead deadline

# Homework deadlines




- Not to scale
- Blue line: If you make an initial submission by the initial submission deadline
- Red line: If you miss the initial submission deadline (which has no slack)

# Preparation for the course

- Course is implementation heavy
  - A lot of coding and experimenting
  - Will work with some large datasets

- Language of choice: Python
- Toolkit of choice:  Pytorch
  - You are welcome to use other languages/toolkits, but the TAs will not be able to help with coding/homework
    - Some support for TensorFlow

- We hope you have gone through
  - Recitation zero
  - HW zero
    - Carries marks

# Additional Logistics

- Discussions:
  - On Piazza
- Compute infrastructure:
  - Everyone gets Amazon tokens
  - Initially a token for $50
  - Can get additional tokens of $50 up to a total of $150

# *This course is not easy*

- A lot of work!

# *This course is not easy*

- A lot of work!
- *A lot of work!!*

# *This course is not easy*

- A lot of work!
- *A lot of work!!*
- ***A lot of work!!!***

# *This course is not easy*

- A lot of work!
- *A lot of work!!*
- ***A lot of work!!!***
- ***A LOT OF WORK!!!!***

Not for chicken!

# *This course is not easy*

- A lot of work!
- *A lot of work!!*
- ***A lot of work!!!***
- ***A LOT OF WORK!!!!***
- *Mastery-based* evaluation
  - Quizzes to test your understanding of topics covered in the lectures
  - HWs to teach you to implement complex networks
    - And optimize them to high degree
- Target: Anyone who gets an "A" in the course is technically ready for a deep learning job

# HW0 / Recitation 0

- Please, please, please, please, please go through the videos for recitation 0, and complete HW0.
  - These are essential for you to gain comfort with the coding require in the following homeworks

- HW1 part 1 also has many components intended to help you *later* in the course
  - So if it seems a bit dense, please bear with it, its worth it
- HW1 is the easiest HW!

# Class Room

- This is the *last* class in this room
- Friday's class is in Posner 151
- Classes from next week will be in GHC 4307
  - Can hold 70 people

# Questions?

- Please post on piazza

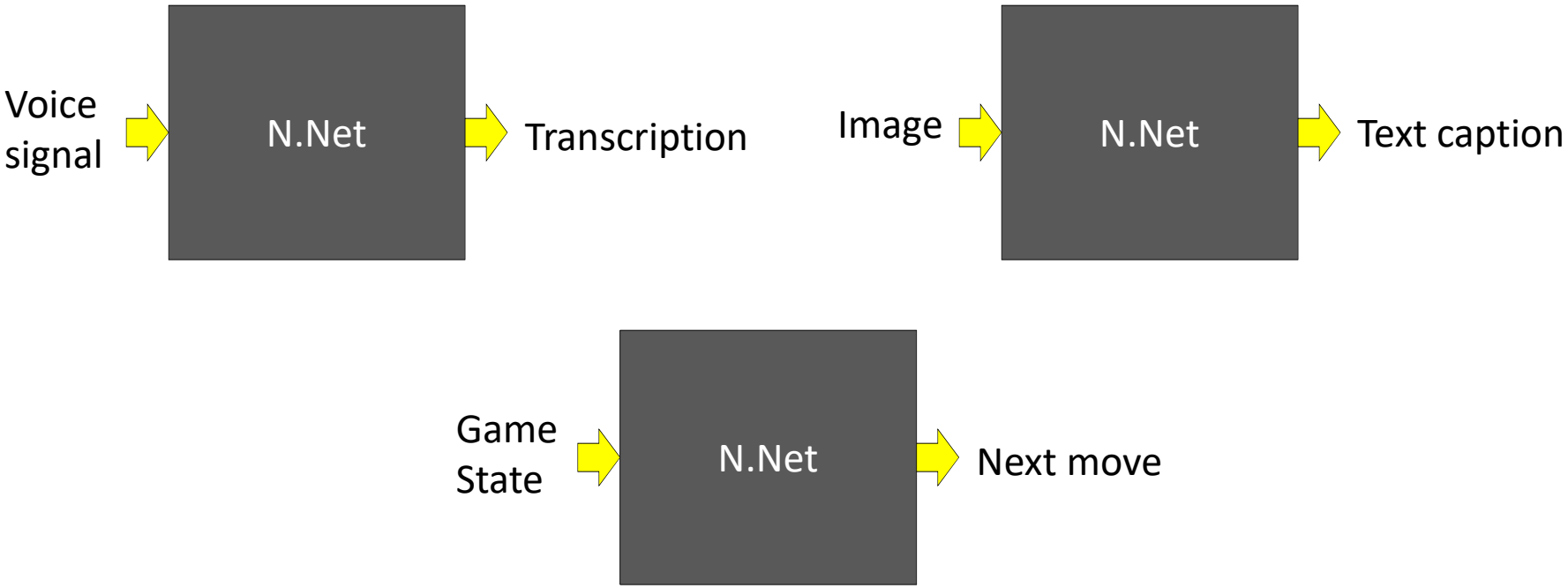
# *Perception:* From Rosenblatt, 1962..

- "Perception, then, emerges as that relatively primitive, partly autonomous, institutionalized, ratiomorphic subsystem of cognition which achieves prompt and richly detailed orientation habitually concerning the vitally relevant, mostly distal aspects of the environment on the basis of mutually vicarious, relatively restricted and stereotyped, insufficient evidence in uncertainty-geared interaction and compromise, seemingly following the highest probability for smallness of error at the expense of the highest frequency of precision. "
  - From "Perception and the Representative Design of Psychological Experiments, " by Egon Brunswik, 1956 (posthumous).

- "That's a simplification. Perception is standing on the sidewalk, watching all the girls go by."
  - From "The New Yorker", December 19, 1959

# Onward..

# So what are neural networks??

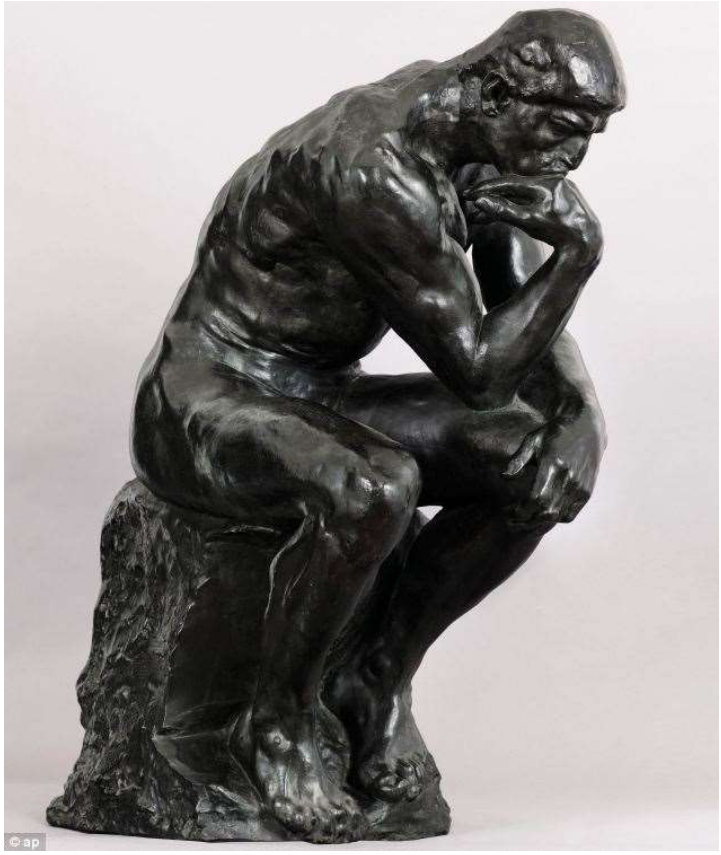



- What are these boxes?

# So what are neural networks??

- It begins with this..

# So what are neural networks??



"The Thinker!"
by Augustin Rodin

- Or even earlier.. with this..

# The magical capacity of humans

- Humans can
  - Learn
  - Solve problems
  - Recognize patterns
  - Create
  - Cogitate
  - …


- Worthy of emulation
- But how do humans “work“?

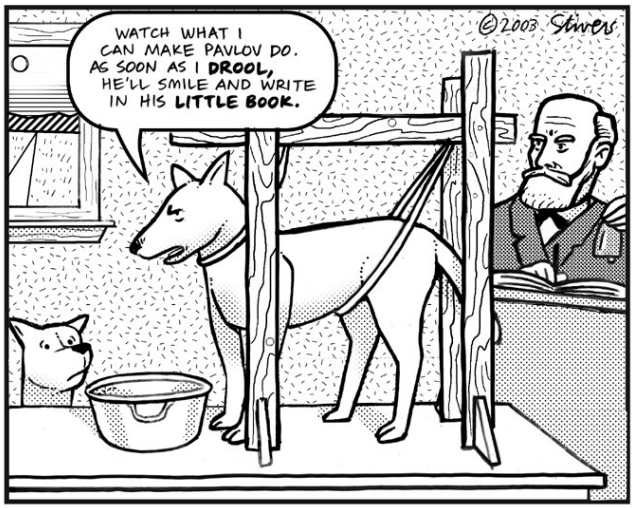
# Cognition and the brain..

- “If the brain was simple enough to be understood - we would be too simple to understand it!”
  - Marvin Minsky

# Early Models of Human Cognition

- Associationism
  - Humans learn through association
- 400BC-1900AD: Plato, David Hume, Ivan Pavlov..

# What are "Associations"




- Lightning is generally followed by thunder
  - Ergo – "hey here's a bolt of lightning, we're going to hear thunder"
  - Ergo – "We just heard thunder; did someone get hit by lightning"?

- Association!

# A little history : Associationism

- Collection of ideas stating a basic philosophy:
  - "Pairs of thoughts become associated based on the organism's past experience"
  - ***Learning*** is a mental process that forms associations between temporally related phenomena

- 360 BC: Aristotle
  - "Hence, too, it is that we hunt through the mental train, excogitating from the present or some other, and from similar or contrary or coadjacent. Through this process reminiscence takes place. For the movements are, in these cases, sometimes at the same time, sometimes parts of the same whole, so that the subsequent movement is already more than half accomplished."
    - In English: *we memorize and rationalize through association*

# Aristotle and Associationism

- Aristotle's four laws of association:
  - *The law of contiguity*. Things or events that occur close together in space or time get linked together
  - *The law of frequency*. The more often two things or events are linked, the more powerful that association.
  - *The law of similarity*. If two things are similar, the thought of one will trigger the thought of the other
  - *The law of contrast*. Seeing or recalling something may trigger the recollection of something opposite.

# A little history : **Associationism**

- More recent associationists (upto 1800s): John Locke, David Hume, David Hartley, James Mill, John Stuart Mill, **Alexander Bain**, Ivan Pavlov
  - Associationist theory of mental processes: *there is only one mental process: the ability to associate ideas*
  - Associationist theory of learning: *cause and effect, contiguity, resemblance*
  - Behaviorism (early 20th century) : *Behavior is learned from repeated associations of actions with feedback*
  - Etc.

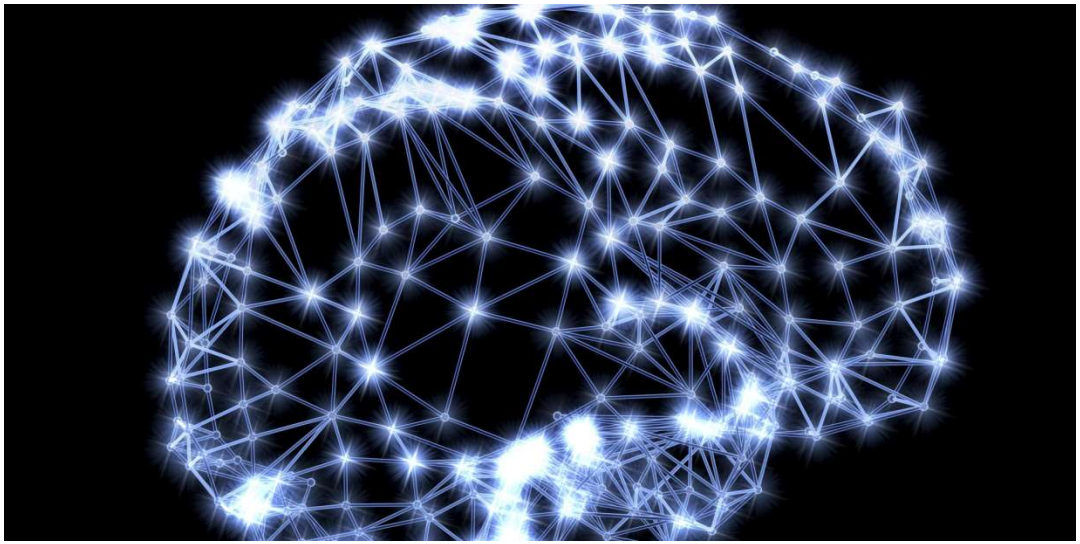
- **But where are the associations stored??**
- **And how?**

# But how do we *store* them?
# Dawn of Connectionism

David Hartley's *Observations on man* (1749)

- We receive input through vibrations and those are transferred to the brain
- Memories could also be small vibrations (called vibratiuncles) in the same regions
- Our brain represents compound or connected ideas by connecting our memories with our current senses
- Current science did not know about neurons

# Observation: ***The Brain***

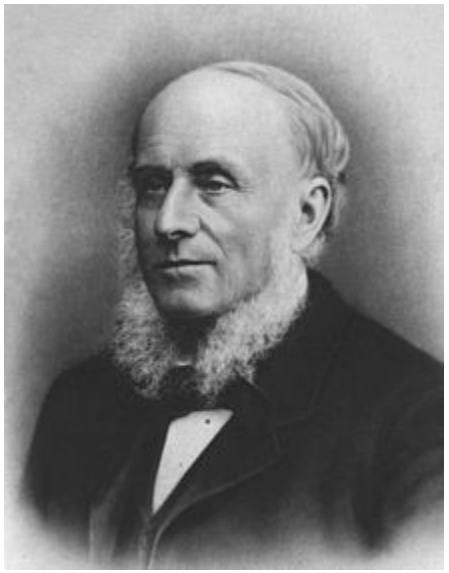


- Mid 1800s:  The brain is a mass of interconnected neurons

# Brain: Interconnected Neurons

- Many neurons connect *in* to each neuron
- Each neuron connects *out* to many neurons

# Enter ***Connectionism***



- Alexander Bain, philosopher, mathematician, logician, linguist, professor
- 1873: The information is in the ***connections***
  - *The mind and body* (1873)

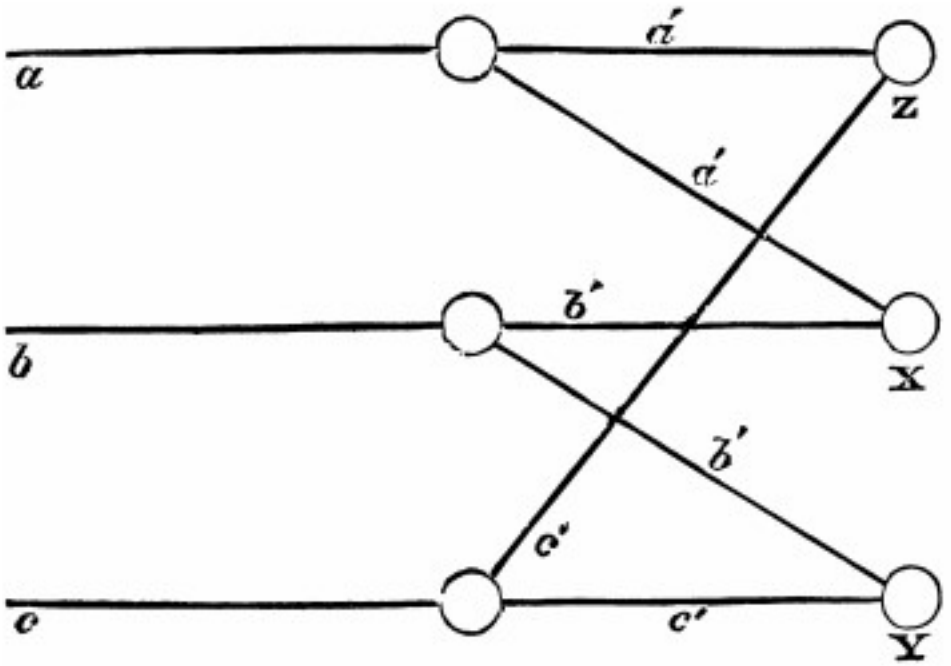
# Enter: **Connectionism**

Alexander Bain (*The senses and the intellect* (1855), *The emotions and the will* (1859), *The mind and body* (1873))

- In complicated words:
  - Idea 1: The "nerve currents" from a memory of an event are the same but reduce from the "original shock"
  - Idea 2: "for every act of memory, … there is a specific grouping, or co-ordination of sensations … *by virtue of specific growths in cell junctions*"

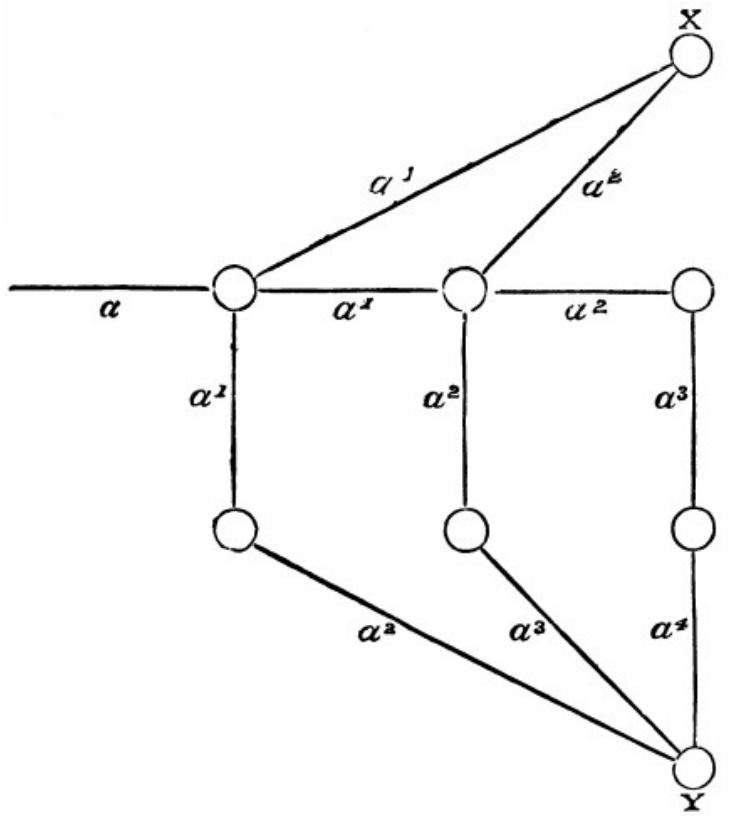
# Bain's Idea 1: **Neural Groupings**

- Neurons excite and stimulate each other
- Different combinations of inputs can result in different outputs



# Bain's Idea 1: **Neural Groupings**

- Different intensities of activation of A lead to the differences in when X and Y are activated




- Even proposed a learning mechanism..

# Bain's Idea 2: **Making Memories**

- "when two impressions concur, or closely succeed one another, the nerve currents find some bridge or place of continuity, better or worse, according to the abundance of nerve matter available for the transition."

- Predicts "Hebbian" learning (three quarters of a century before Hebb!)

# Bain's Doubts

- "***The fundamental cause of the trouble is that in the modern world the stupid are cocksure while the intelligent are full of doubt.***"
  - Bertrand Russell

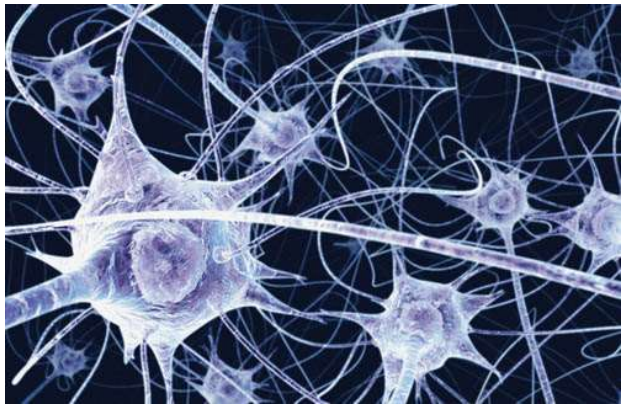
- In 1873, Bain postulated that there must be one million neurons and 5 billion connections relating to 200,000 "acquisitions"

- In 1883, Bain was concerned that he hadn't taken into account the number of "partially formed associations" and the number of neurons responsible for recall/learning

- By the end of his life (1903), recanted all his ideas!
  - Too complex; the brain would need too many neurons and connections

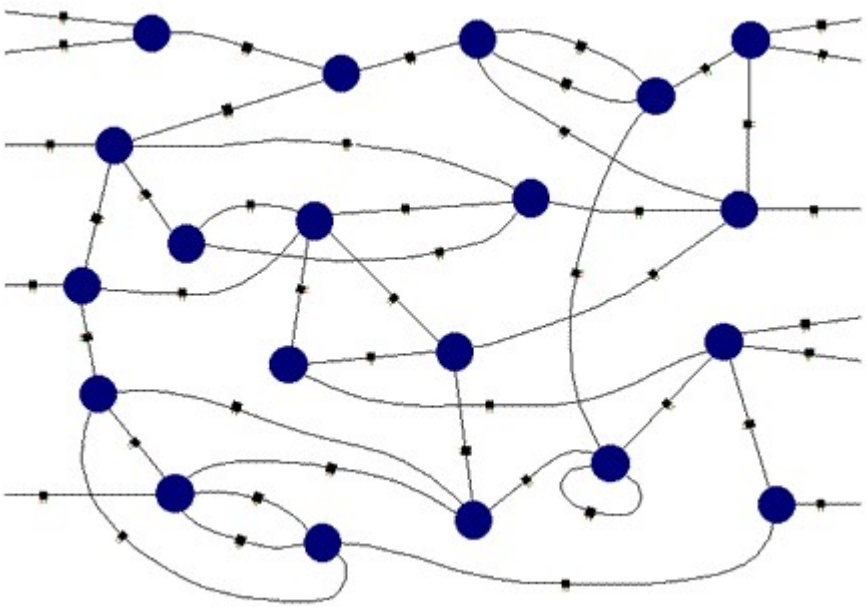
# *Connectionism lives on..*

- The human brain is a connectionist machine
  - Bain, A. (1873). *Mind and body. The theories of their relation.* London: Henry King.
  - Ferrier, D. (1876). *The Functions of the Brain.* London: Smith, Elder and Co



- Neurons connect to other neurons. The processing/capacity of the brain is a function of these connections
- Connectionist machines emulate this structure

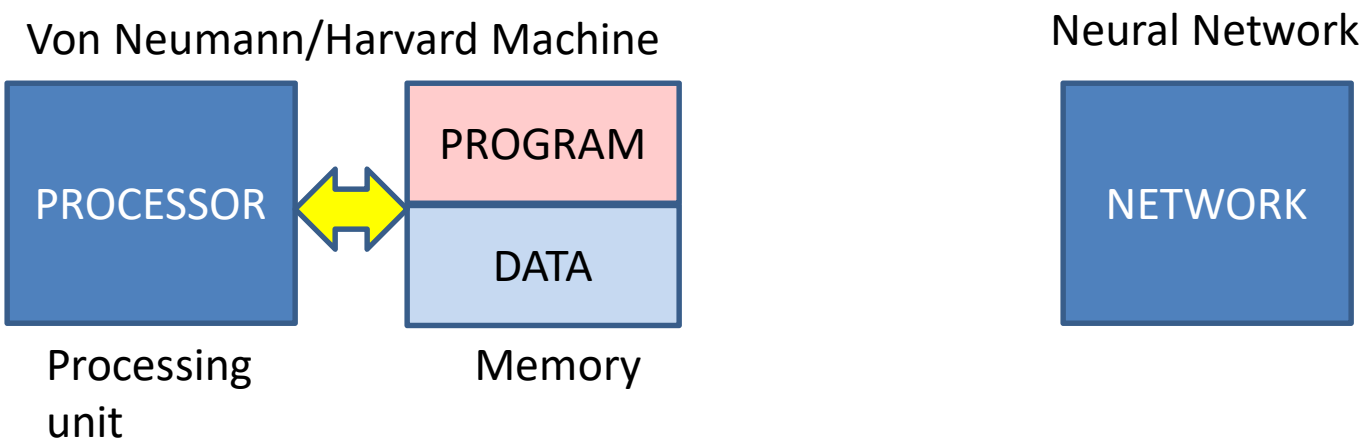
# Connectionist Machines



- Network of processing elements
- **All world knowledge is stored in the *connections* between the elements**

# Connectionist Machines

- Neural networks are *connectionist* machines
  - As opposed to Von Neumann Machines

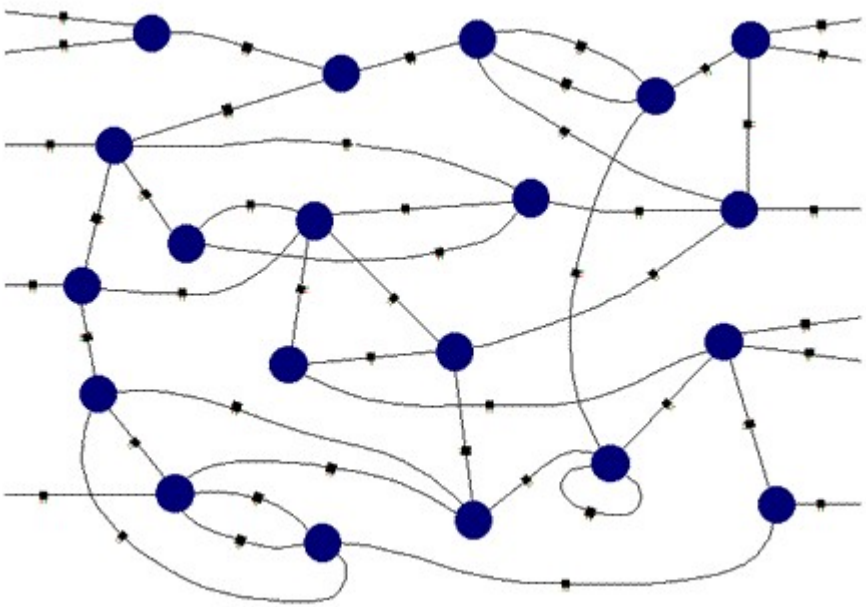



- The machine has many non-linear processing units
  - The program is the connections between these units
    - Connections may also define memory

# Recap

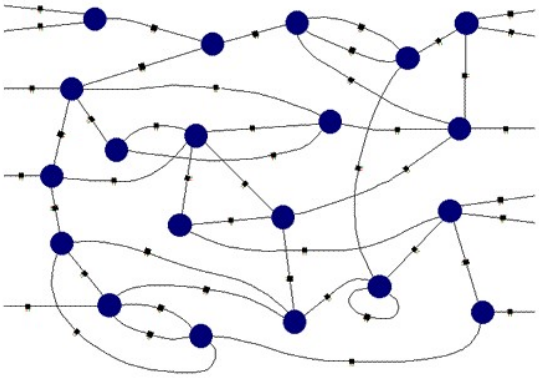
- Neural network based AI has taken over most AI tasks
- Neural networks originally began as computational models of the brain
  - Or more generally, models of cognition
- The earliest model of cognition was *associationism*
- The more recent model of the brain is *connectionist*
  - Neurons connect to neurons
  - The workings of the brain are encoded in these connections
- Current neural network models are *connectionist machines*

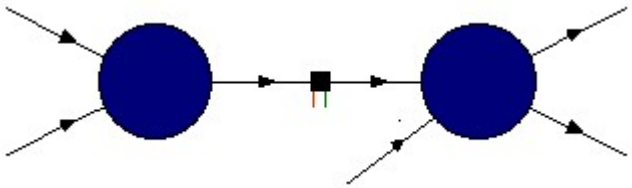
# Connectionist Machines



- Network of processing elements
- All world knowledge is stored in the *connections* between the elements
- *Multiple* connectionist paradigms proposed..

# Turing's Connectionist Machines





- Basic model: A-type machines
  - Networks of NAND gates
- Connectionist model: B-type machines (1948)
  - Connection between two units has a "modifier"
  - If the green line is on, the signal sails through
  - If the red is on, the output is fixed to 1
  - "Learning" – figuring out how to manipulate the coloured wires
    - Done by an A-type machine

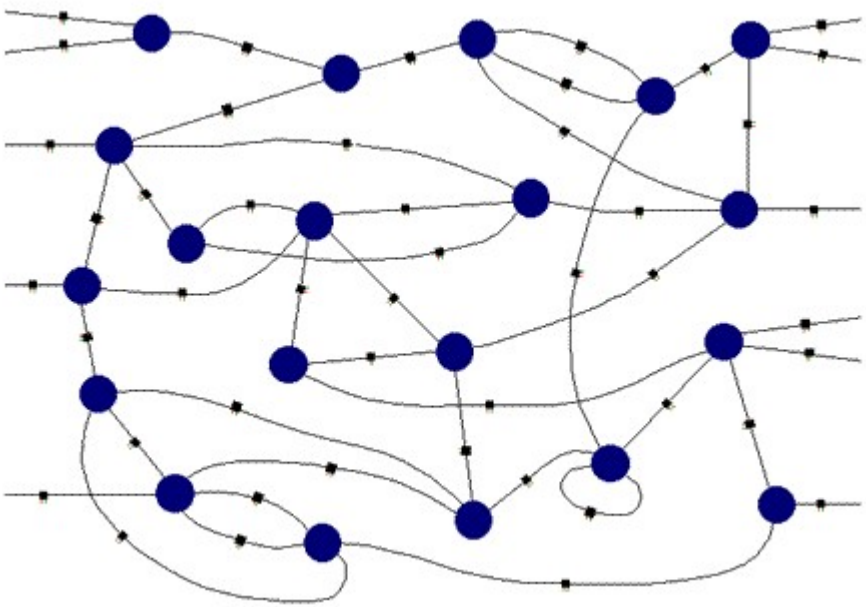
# Connectionist paradigms: PDP Parallel Distributed Processing

- Requirements for a PDP system
  (Rumelhart, Hinton, McClelland, ‘86; quoted from Medler, ‘98)
  - A set of processing units
  - A state of activation
  - An output function for each unit
  - A pattern of connectivity among units
  - A propagation rule for propagating patterns of activities through the network of connectivities
  - An activation rule for combining the inputs impinging on a unit with the current state of that unit to produce a new level of activation for the unit
  - A learning rule whereby patterns of connectivity are modified by experience
  - An environment within which the system must operate

# Connectionist Systems

- Requirements for a connectionist system (Bechtel and Abrahamson, 91)
  - The connectivity of units
  - The activation function of units
  - The nature of the learning procedure that modifies the connections between units, and
  - How the network is interpreted semantically

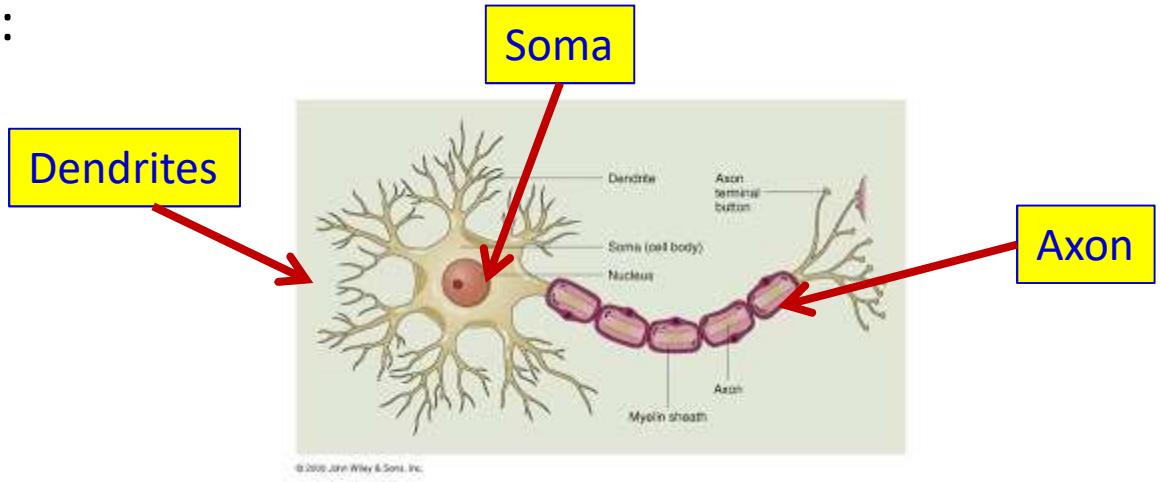
# Connectionist Machines



- Network of processing elements
  - All world knowledge is stored in the *connections* between the elements
- *But what are the individual elements?*

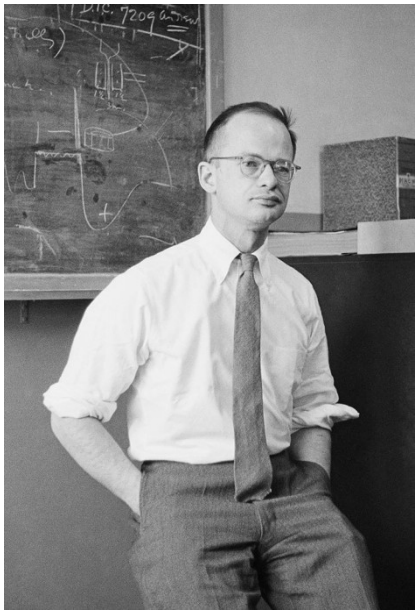
# Modelling the brain

- What are the units?
- A neuron:



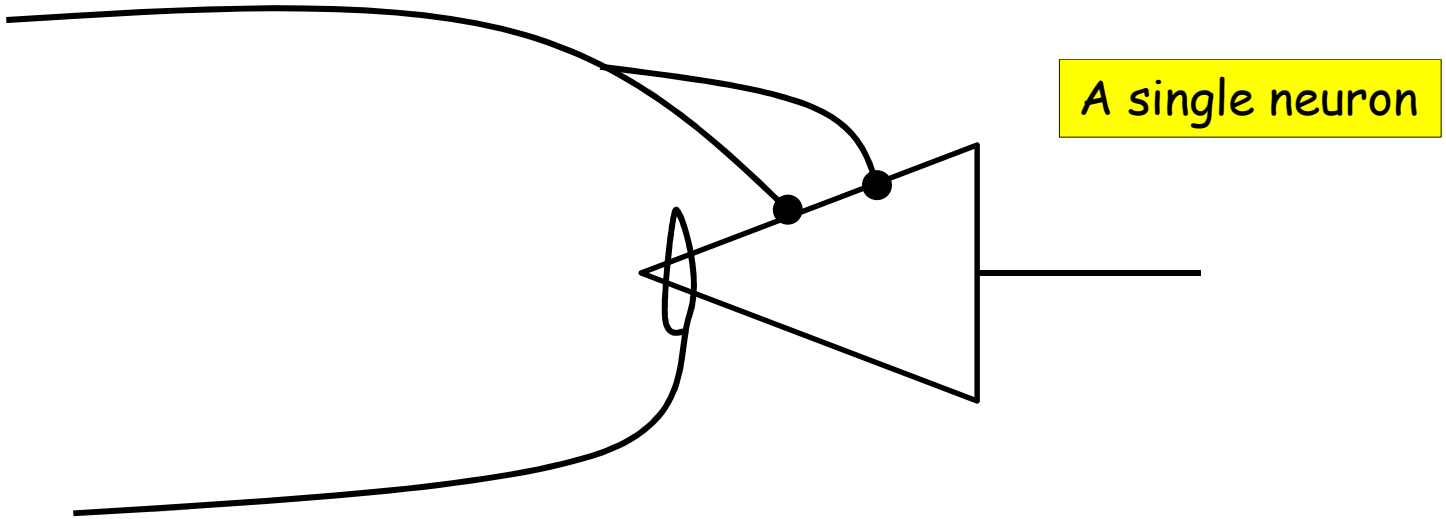

- Signals come in through the dendrites into the Soma
- A signal goes out via the axon to other neurons
  - Only one axon per neuron
- Factoid that may only interest me: Adult neurons do not undergo cell division

# McCullough and Pitts



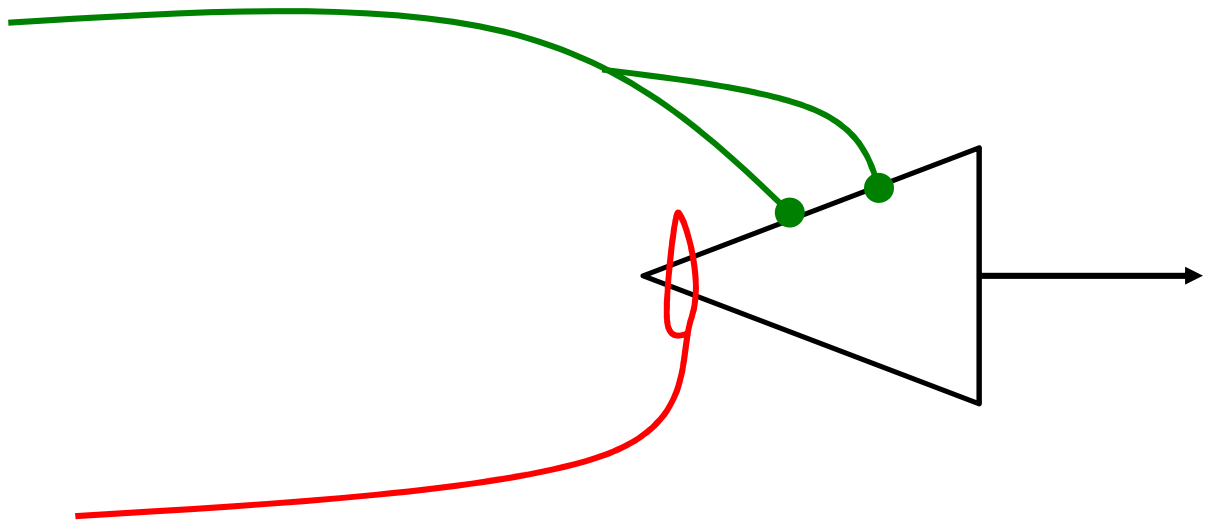
- The Doctor and the Hobo..
  - Warren McCulloch:  Neurophysician
  - Walter Pitts: Homeless wannabe logician who arrived at his door

# The McCulloch and Pitts model




- A mathematical model of a neuron
  - McCulloch, W.S. & Pitts, W.H. (1943). A Logical Calculus of the Ideas Immanent in Nervous Activity, Bulletin of Mathematical Biophysics, 5:115-137, 1943
    - Pitts was only 20 years old at this time

# Synaptic Model



- ***Excitatory synapse:*** Transmits weighted input to the neuron
- ***Inhibitory synapse:*** Any signal from an inhibitory synapse forces output to zero
  - The activity of any inhibitory synapse absolutely prevents excitation of the neuron at that time.
    - Regardless of other inputs

# McCullouch and Pitts model

- Made the following assumptions
  - The activity of the neuron is an ‘‘all-or-none’’ process
  - 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
  - The only significant delay within the nervous system is synaptic delay
  - The activity of any inhibitory synapse absolutely prevents excitation of the neuron at that time
  - The structure of the net does not change with time

Simple "networks" of neurons can perform Boolean operations

# Boolean Gates

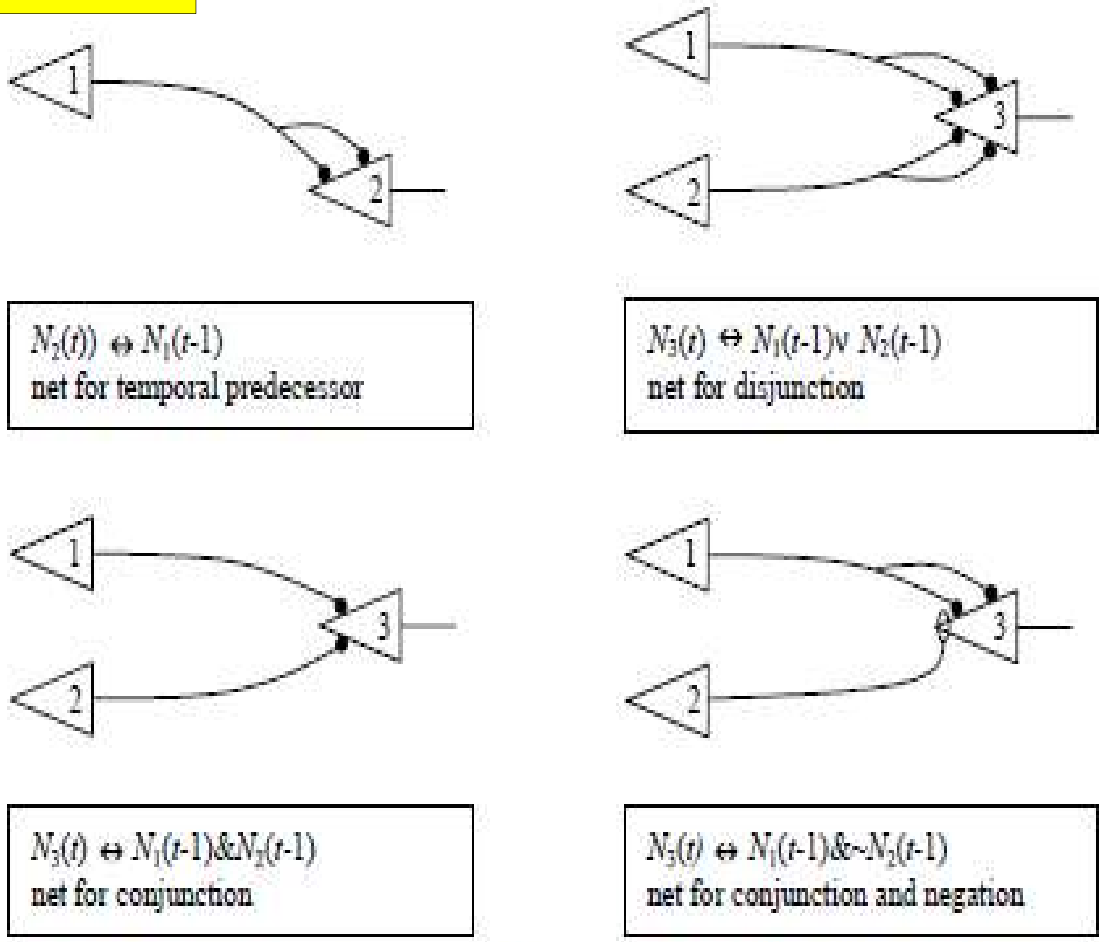



*Figure 1.* Diagrams of McCulloch and Pitts nets. In order to send an output pulse, each neuron must receive two excitory inputs and no inhibitory inputs. Lines ending in a dot represent excitatory connections; lines ending in a hoop represent inhibitory connections.

# Complex Percepts & Inhibition in action

They can even create illusions of "perception"

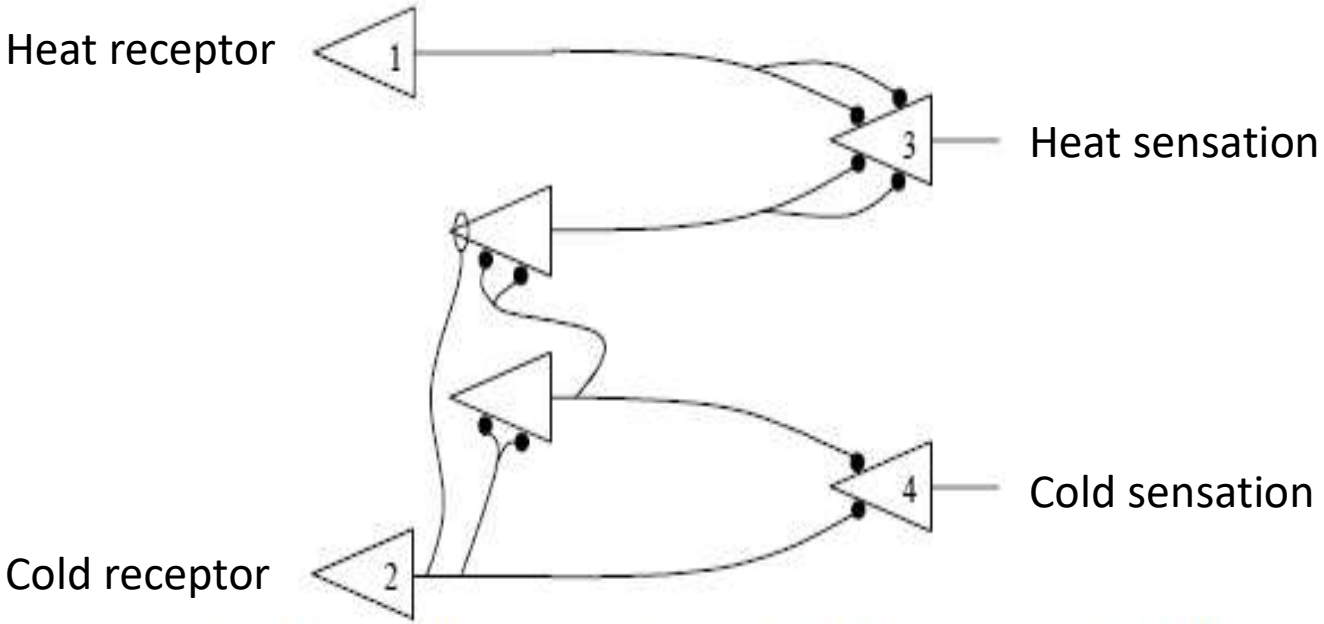
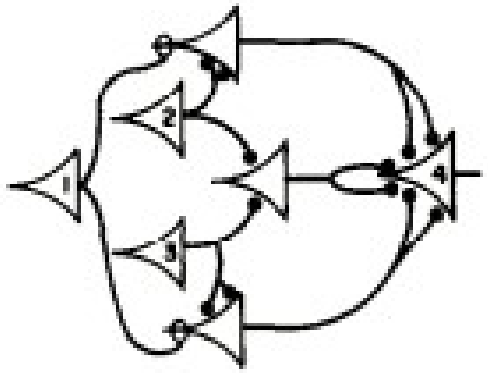


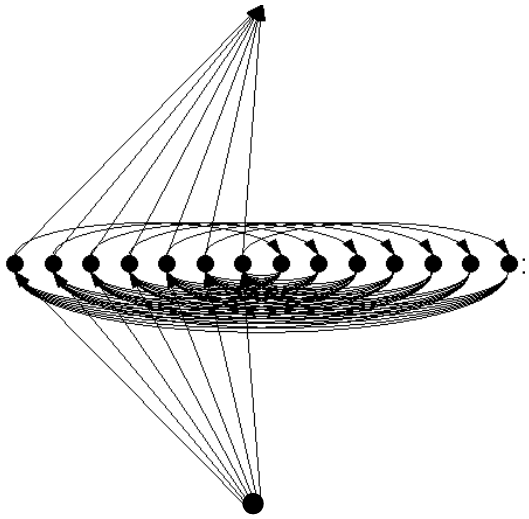

*Figure 2.* Net explaining the heat illusion. Neuron 3 (heat sensation) fires if and only if it receives two inputs, represented by the lines terminating on its body. This happens when either neuron 1 (heat reception) fires or neuron 2 (cold reception) fires once and then immediately stops firing. When neuron 2 fires twice in a row, the intermediate (unnumbered) neurons excite neuron 4 rather than neuron 3, generating a sensation of cold.

# McCulloch and Pitts Model

- Could compute arbitrary Boolean propositions
  - Since any Boolean function can be emulated, any Boolean function can be composed



- Models for *memory*
  - Networks with loops can "remember"
    - We'll see more of this later
  - Lawrence Kubie (1930): Closed loops in the central nervous system explain memory



# Criticisms

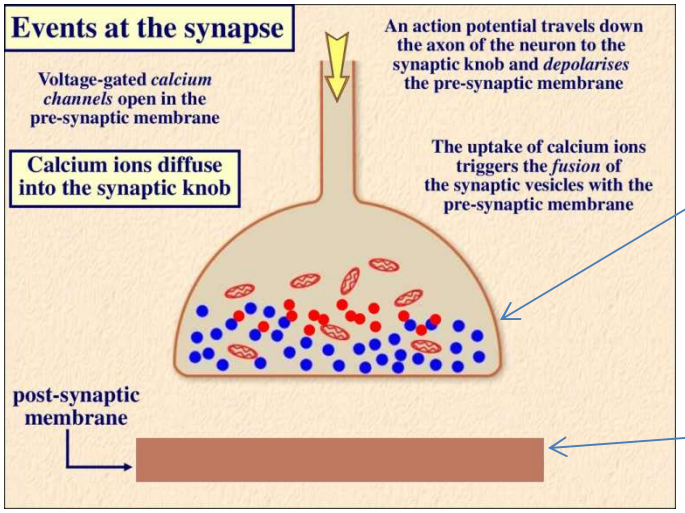
- They claimed that their nets
  - should be able to compute a small class of functions
  - also if tape is provided their nets can compute a richer class of functions.
    - additionally they will be equivalent to Turing machines
    - Dubious claim that they're Turing complete
  - They didn't prove any results themselves

- Didn't provide a learning mechanism..

# Donald Hebb

- "Organization of behavior", 1949
- A learning mechanism:
  - "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."
    - As *A* repeatedly excites *B*, its *ability* to excite *B* improves
  - *Neurons that fire together wire together*

# Hebbian Learning




Axonal connection from neuron X

Dendrite of neuron Y

- If neuron $x_i$ repeatedly triggers neuron $y$, the synaptic knob connecting $x_i$ to $y$ gets larger
- In a mathematical model:

$$w_i = w_i + \eta x_i y$$

  - Weight of $i^{\text{th}}$ neuron's input to output neuron $y$
- This simple formula is actually the basis of many learning algorithms in ML

# Hebbian Learning

- **Fundamentally unstable**
  - Stronger connections will enforce themselves
  - No notion of "competition"
  - No *reduction* in weights
  - Learning is unbounded

- Number of later modifications, allowing for weight normalization, forgetting etc.
  - E.g. Generalized Hebbian learning, aka Sanger's rule

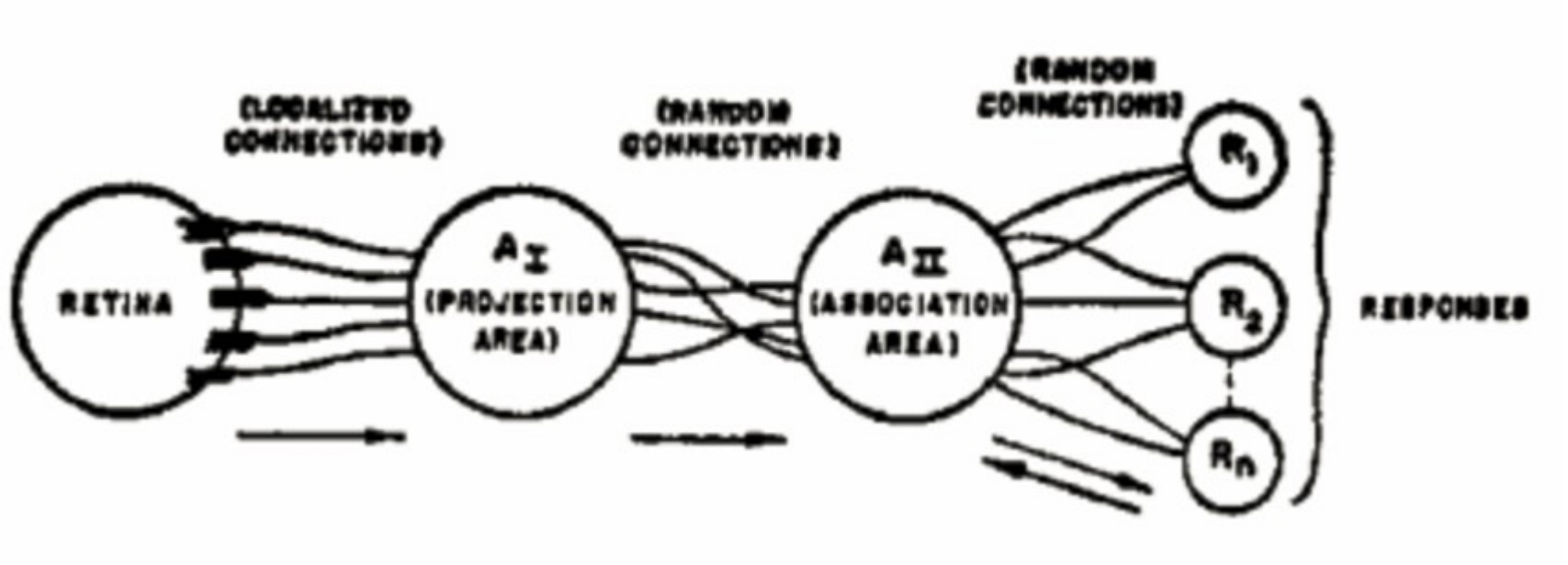
$$w_{ij} = w_{ij} + \eta y_j \left( x_i - \sum_{k=1}^{j} w_{ik} y_k \right)$$

  - The contribution of an input is incrementally *distributed* over multiple outputs..

# A better model

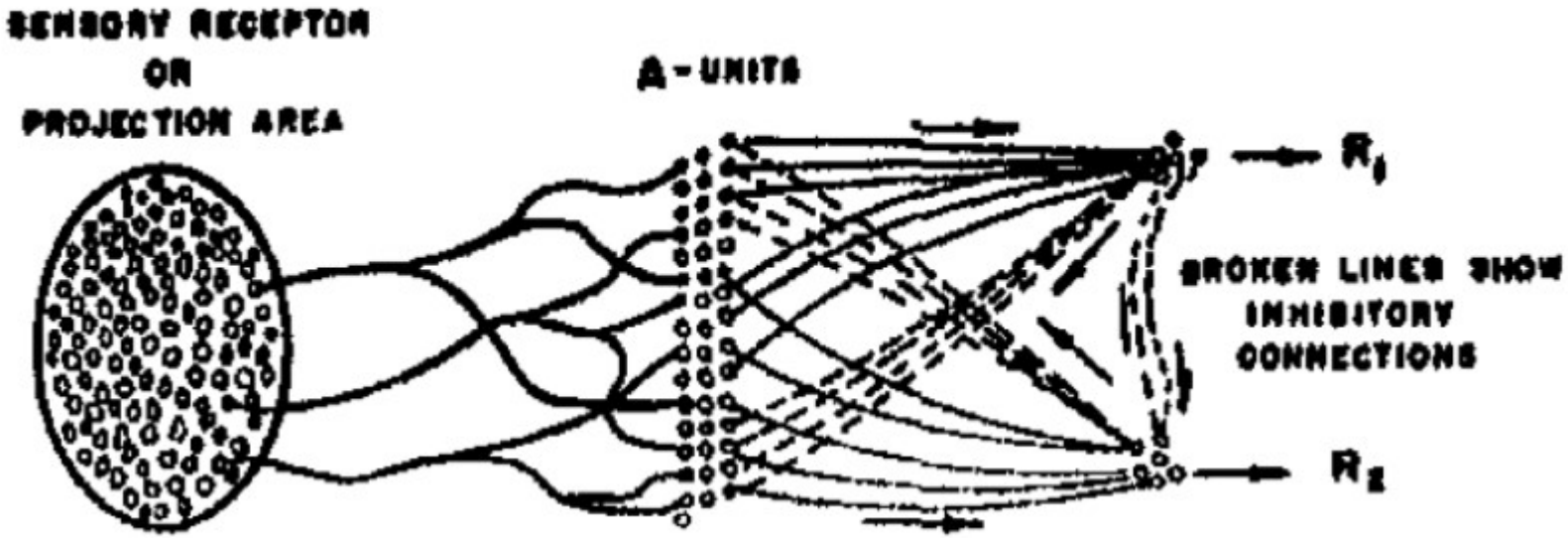

- Frank Rosenblatt
  - Psychologist, Logician
  - Inventor of the solution to everything, aka the Perceptron (1958)

# Rosenblatt's perceptron



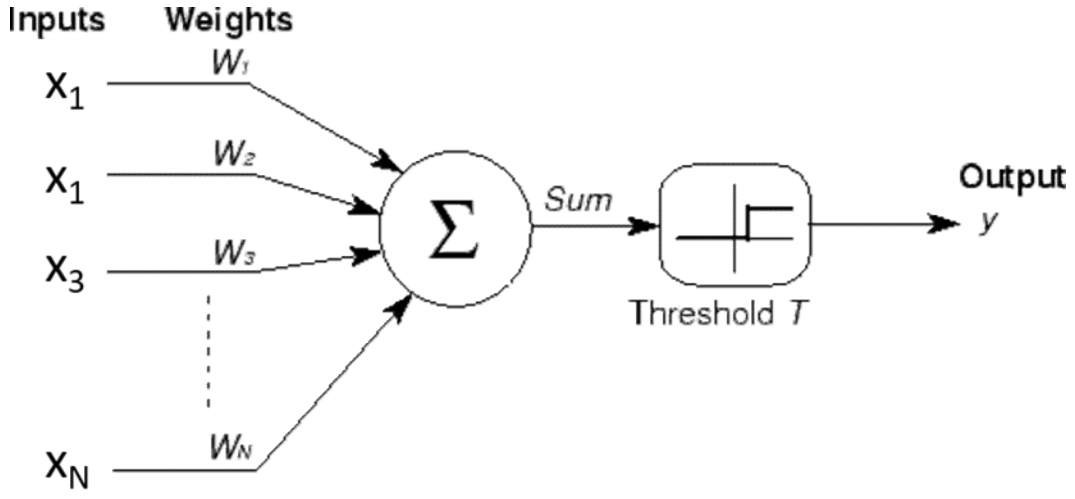

- Original perceptron model
  - Groups of sensors (S) on retina combine onto cells in association area A1
  - Groups of A1 cells combine into Association cells A2
  - Signals from A2 cells combine into response cells R
  - All connections may be excitatory or inhibitory

# Rosenblatt's perceptron




- Even included feedback between A and R cells
  - Ensures mutually exclusive outputs

# Simplified mathematical model

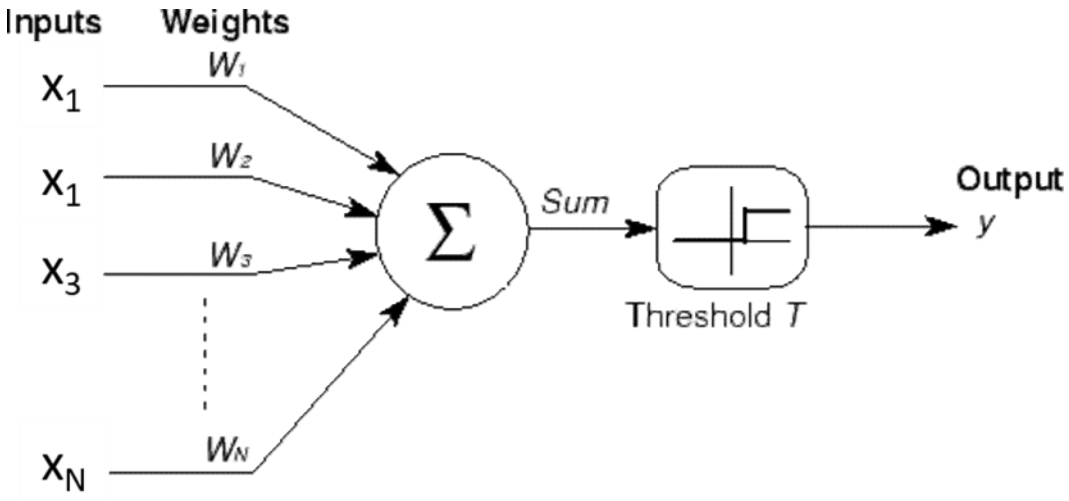



- Number of inputs combine linearly
  - Threshold logic: Fire if combined input exceeds threshold

$$Y = \begin{cases} 1 \;\; if \;\; \sum_i w_i x_i - T > 0 \\ 0 \qquad\qquad\qquad else \end{cases}$$

# His "Simple" Perceptron

- Originally assumed could represent *any* Boolean circuit and perform any logic
  - "*the embryo of an electronic computer that [the Navy] expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence*," New York Times (8 July) 1958
  - "*Frankenstein Monster Designed by Navy That Thinks,*" Tulsa, Oklahoma Times 1958



# Also provided a learning algorithm

$$\mathbf{w} = \mathbf{w} + \eta(d(\mathbf{x}) - y(\mathbf{x}))\mathbf{x}$$

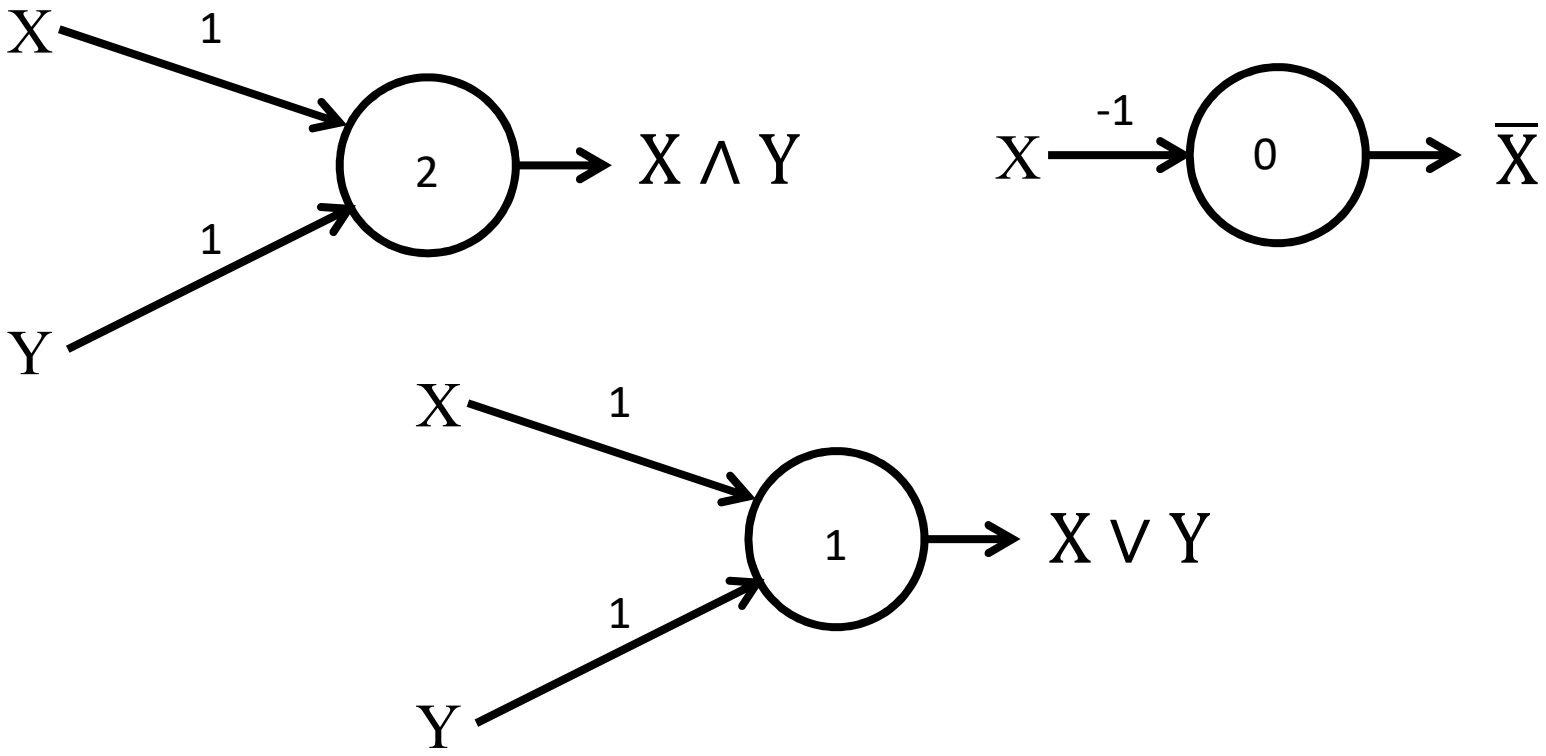
Sequential Learning:

$d(x)$ is the desired output in response to input $x$

$y(x)$ is the actual output in response to $x$

- Boolean tasks
- Update the weights whenever the perceptron output is wrong
- Proved convergence

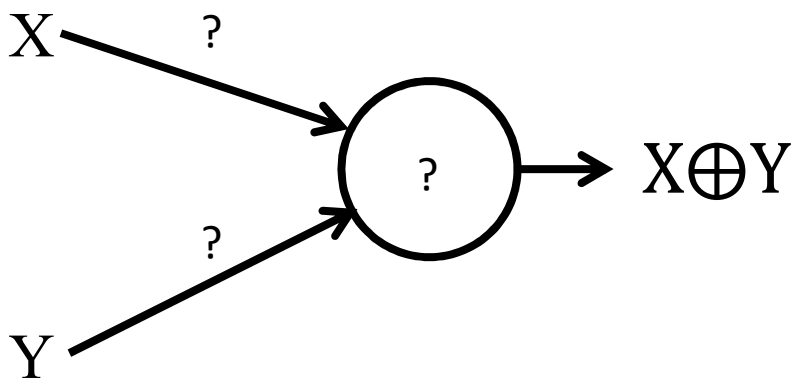
# Perceptron




- Easily shown to mimic any Boolean gate
- But…

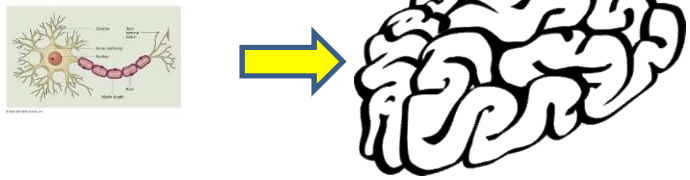
# Perceptron

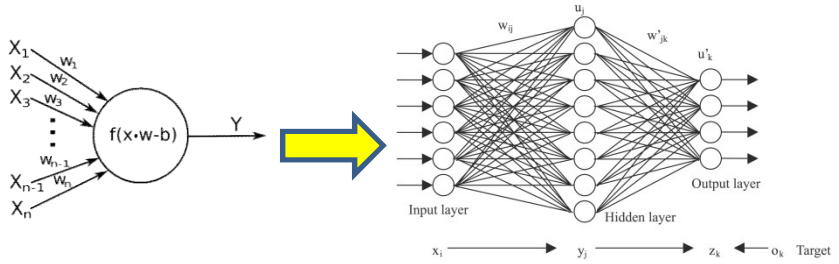
No solution for XOR!
*Not* universal!




- Minsky and Papert, 1968

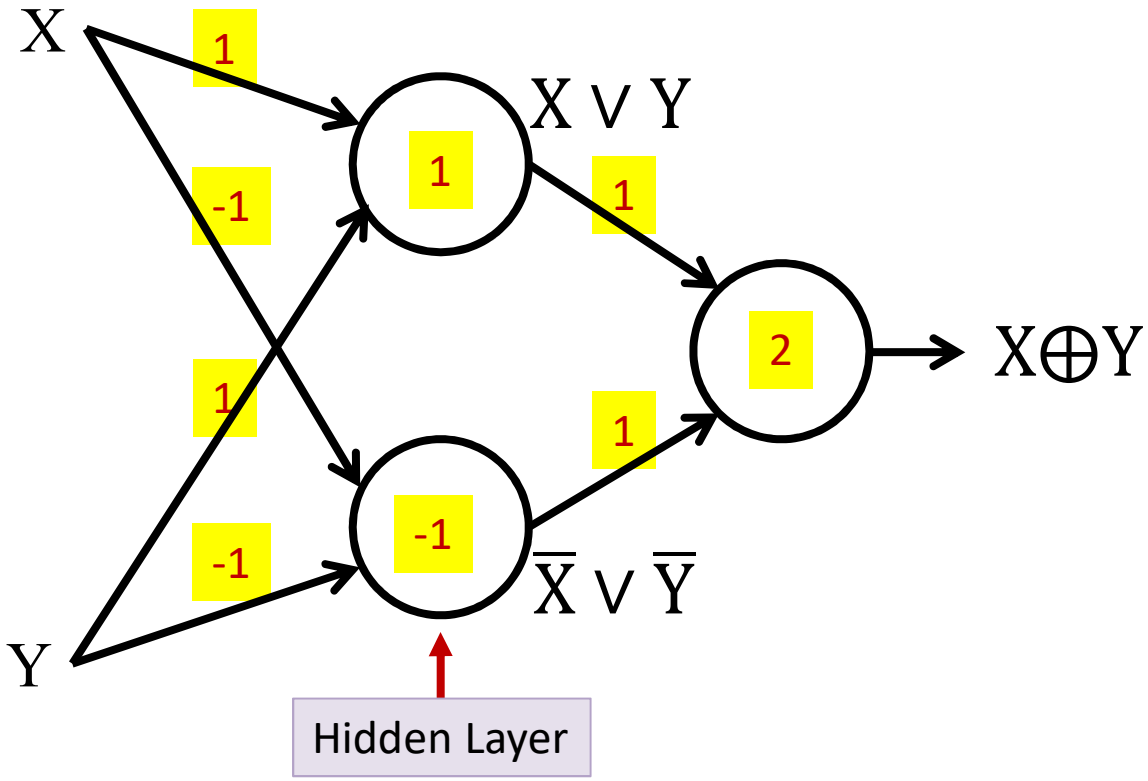
# A single neuron is not enough






- Individual elements are weak computational elements
  - Marvin Minsky and Seymour Papert, 1969, *Perceptrons: An Introduction to Computational Geometry*
- *Networked* elements are required

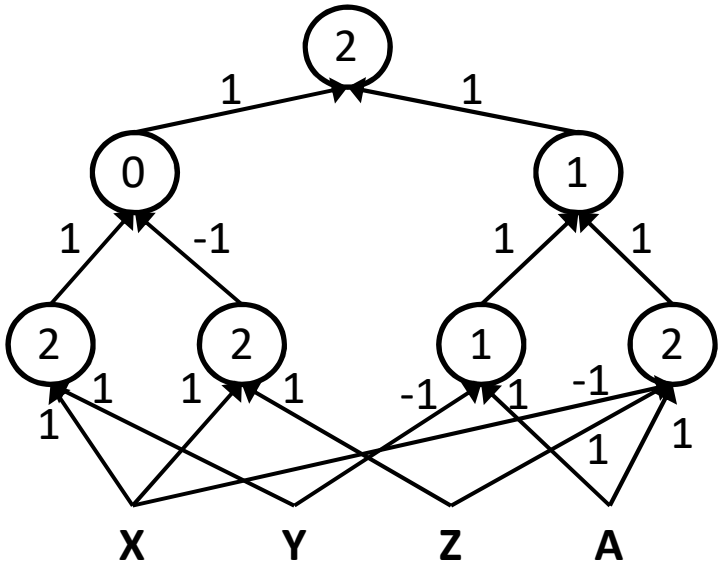
# Multi-layer Perceptron!




- XOR
  - The first layer is a "hidden" layer
  - Also originally suggested by Minsky and Paper 1968

# A more generic model

$$((A\&\bar{X}\&Z)|(A\&\bar{Y}))\&((X\ \&\ Y)|\overline{(X\&Z)})$$

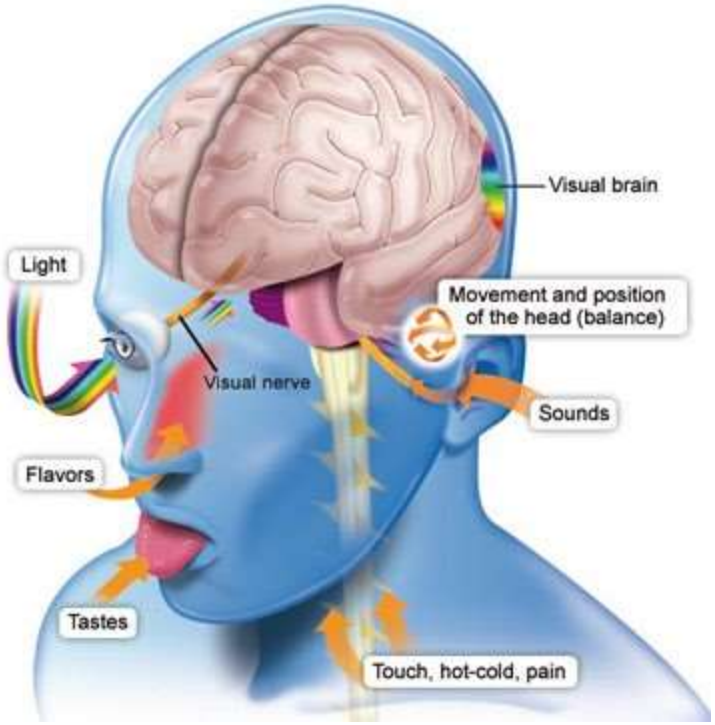



- A "multi-layer" perceptron
- Can compose arbitrarily complicated Boolean functions!
  - In cognitive terms: Can compute arbitrary Boolean functions over sensory input
  - More on this in the next class

# Story so far

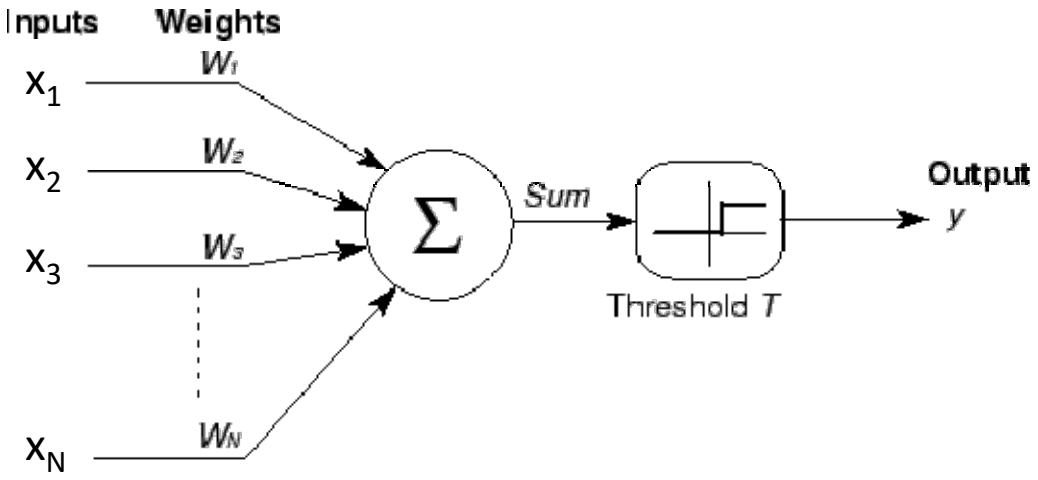
- Neural networks began as computational models of the brain
- Neural network models are *connectionist machines*
  - The comprise networks of neural units
- McCullough and Pitt model: Neurons as Boolean threshold units
  - Models the brain as performing propositional logic
  - But no learning rule
- Hebb's learning rule: Neurons that fire together wire together
  - Unstable
- Rosenblatt's perceptron : A variant of the McCulloch and Pitt neuron with a provably convergent learning rule
  - But individual perceptrons are limited in their capacity (Minsky and Papert)
- Multi-layer perceptrons can model arbitrarily complex Boolean functions

# But our brain is not Boolean



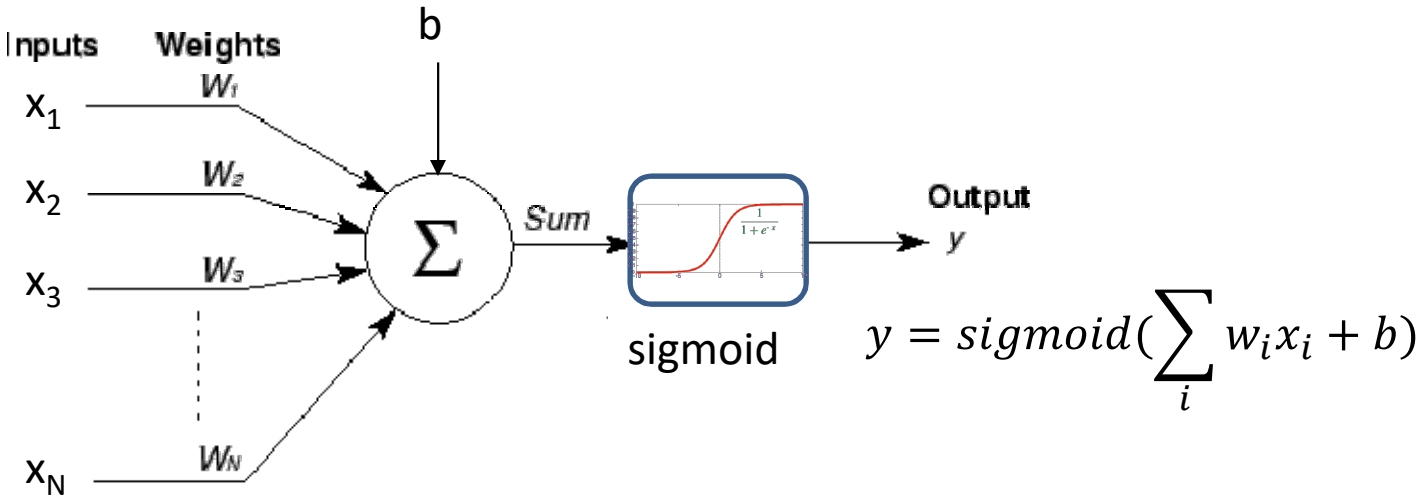

- We have real inputs
- We make non-Boolean inferences/predictions

# The perceptron with *real* inputs



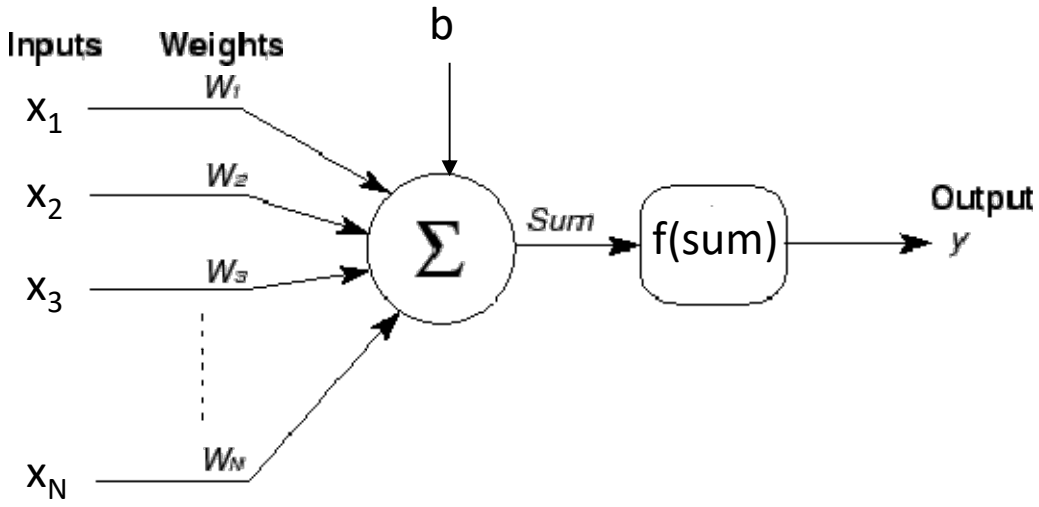

- $x_1 \ldots x_N$ are real valued
- $w_1 \ldots w_N$ are real valued
- Unit "fires" if weighted input exceeds a threshold

# The perceptron with *real* inputs and a real *output*



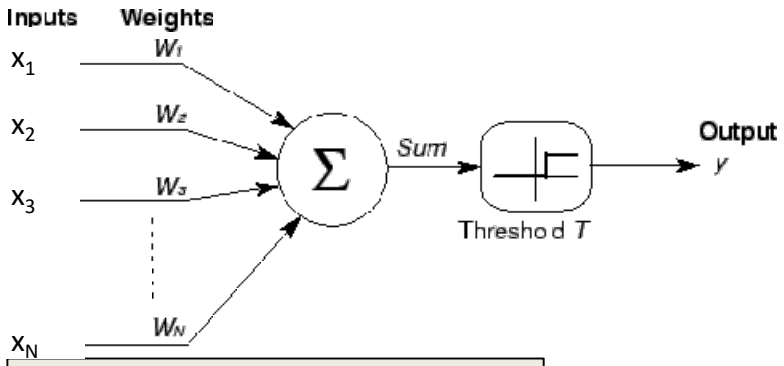

- $x_1 \ldots x_N$ are real valued
- $w_1 \ldots w_N$ are real valued
- The output $y$ can also be real valued
  - Sometimes viewed as the "probability" of firing

# The "real" valued perceptron

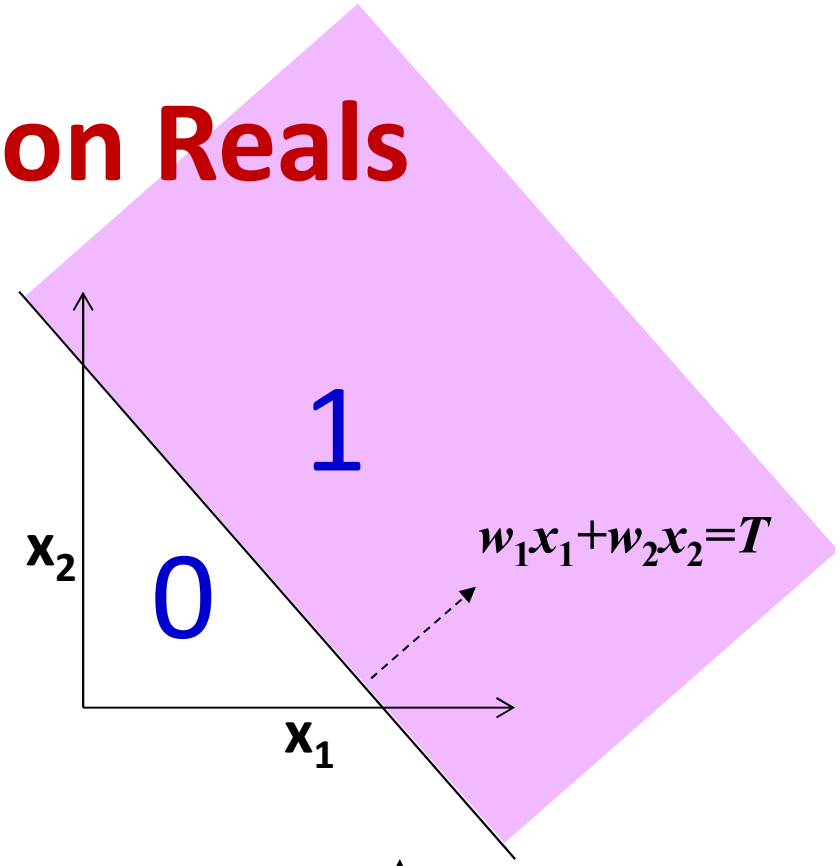


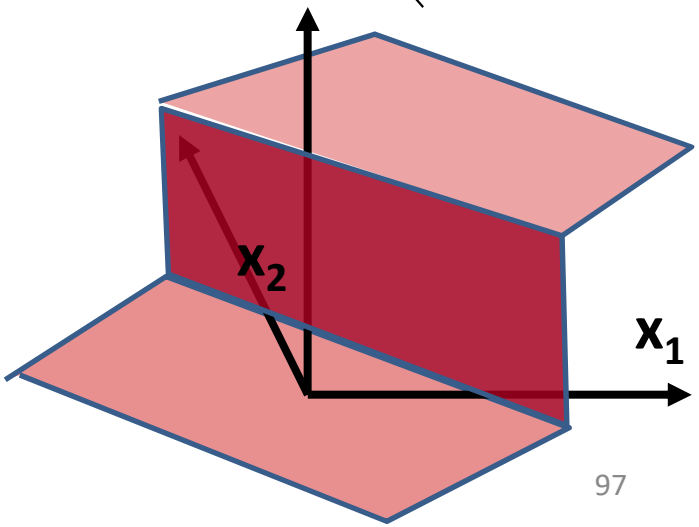

- Any real-valued "activation" function may operate on the weighted-sum input
  - We will see several later
  - Output will be real valued
- The perceptron maps real-valued inputs to real-valued outputs
- *Is useful to continue assuming Boolean outputs though, for interpretation*

# A Perceptron on Reals



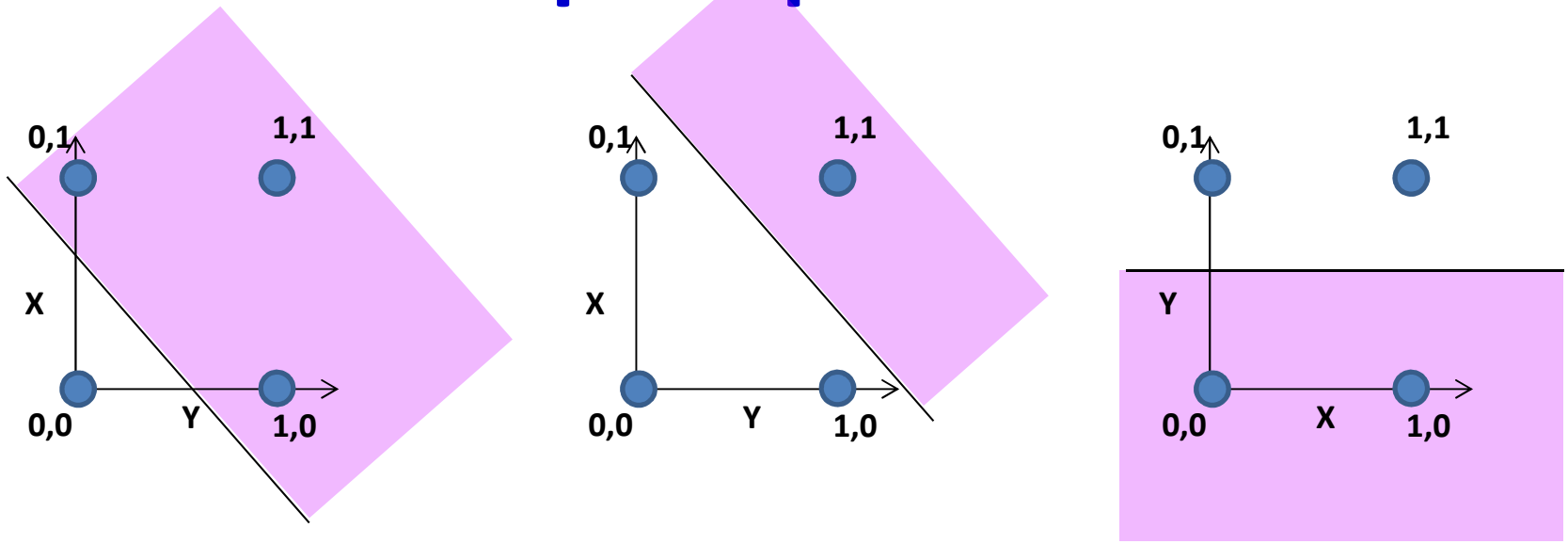

$$y = \begin{cases} 1 \; if \; \sum_i w_i x_i \geq T \\ 0 \; else \end{cases}$$



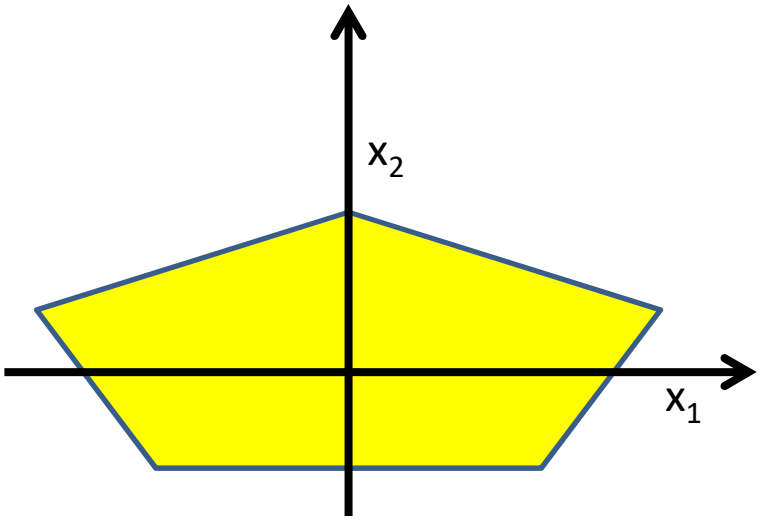

- A perceptron operates on *real*-valued vectors
  - **This is a *linear classifier***

# Boolean functions with a real perceptron




- Boolean perceptrons are also linear classifiers
  - Purple regions have output 1 in the figures
  - What are these functions
  - Why can we not compose an XOR?

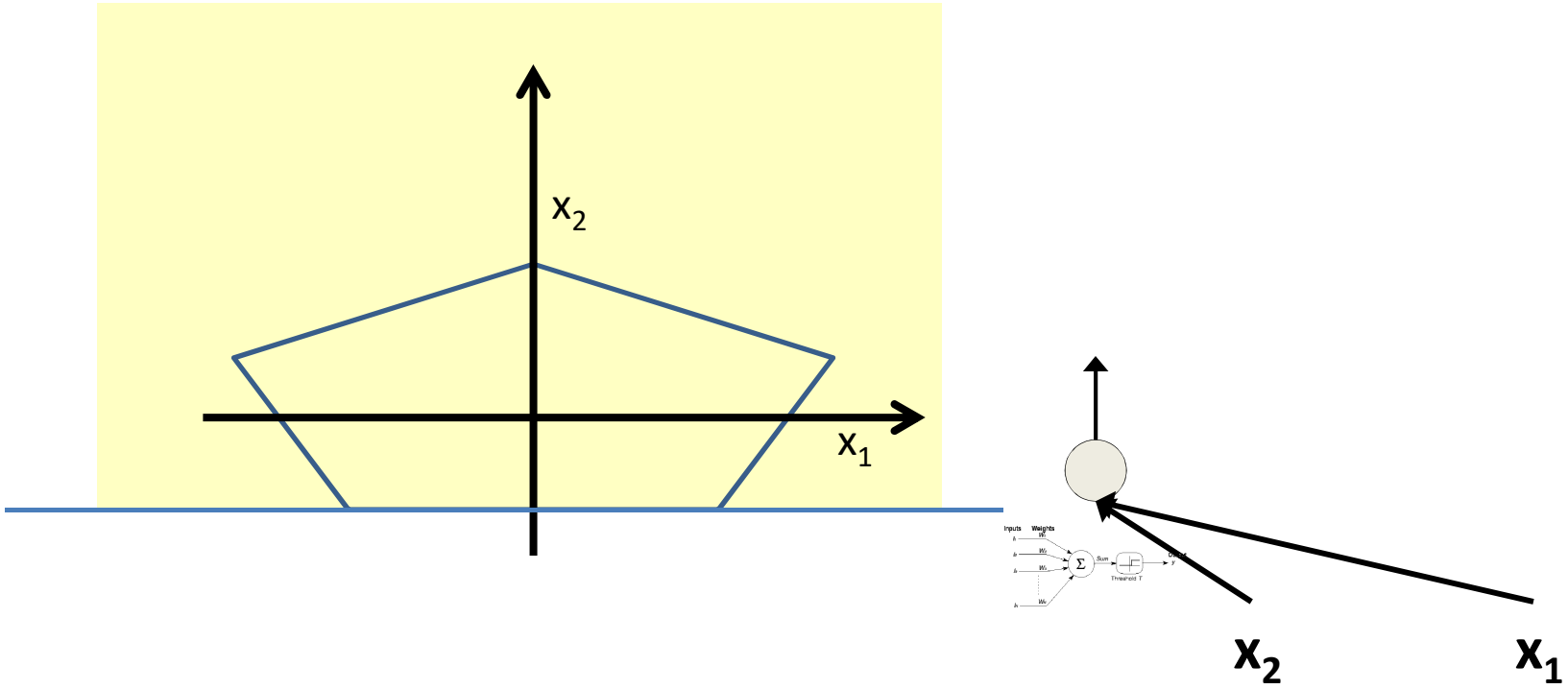
# Composing complicated "decision" boundaries




Can now be composed into "networks" to compute arbitrary classification "boundaries"

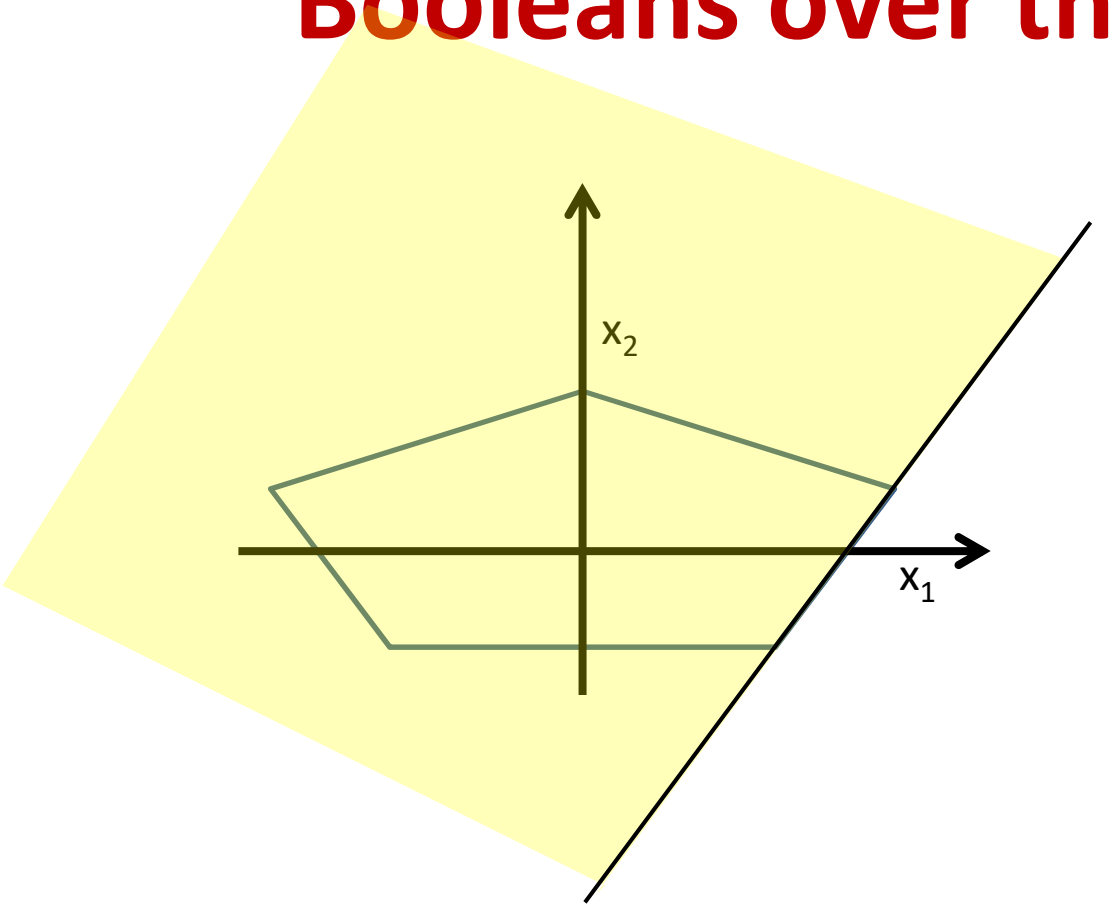
- Build a network of units with a single output that fires if the input is in the coloured area

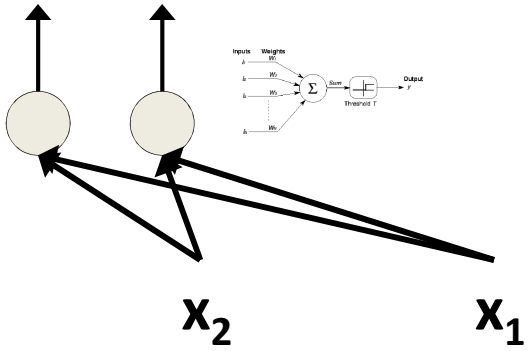
# Booleans over the reals




- The network must fire if the input is in the coloured area

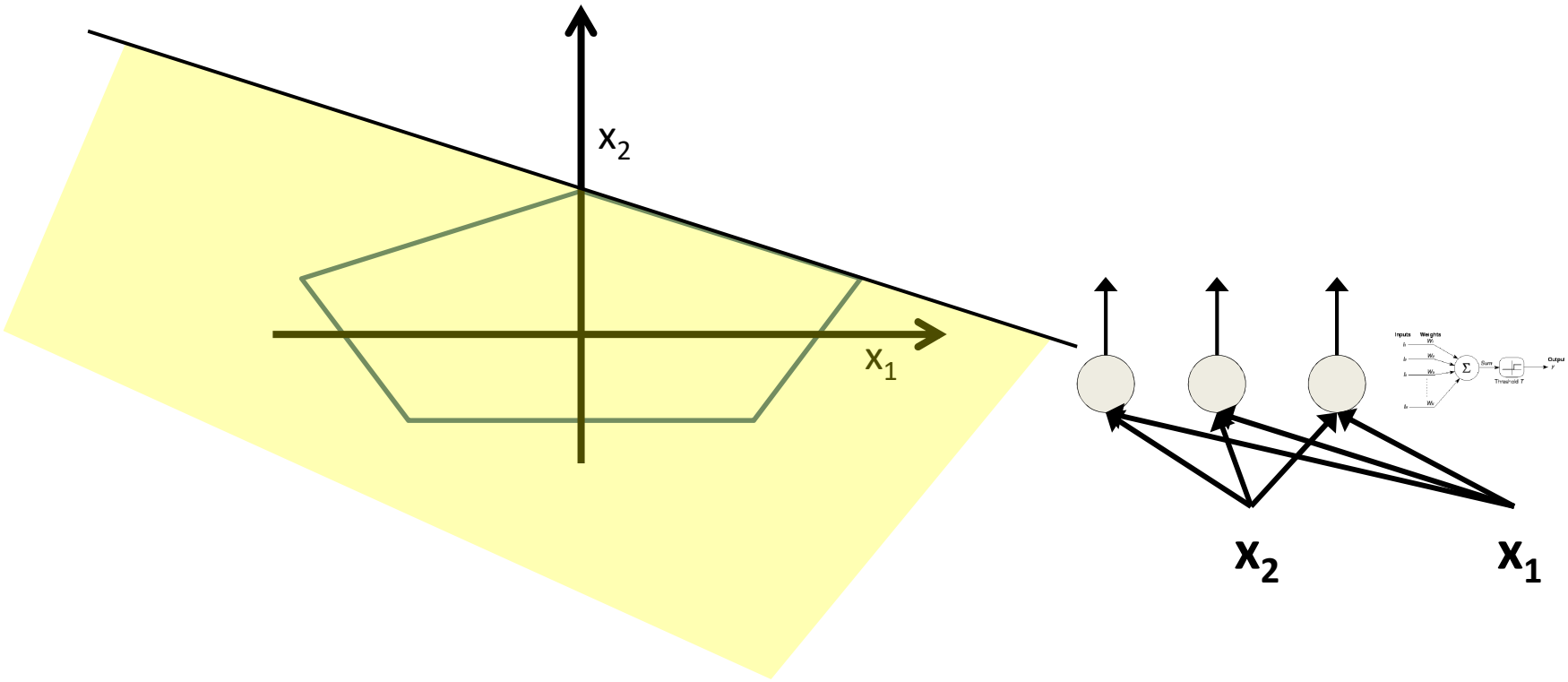
# Booleans over the reals



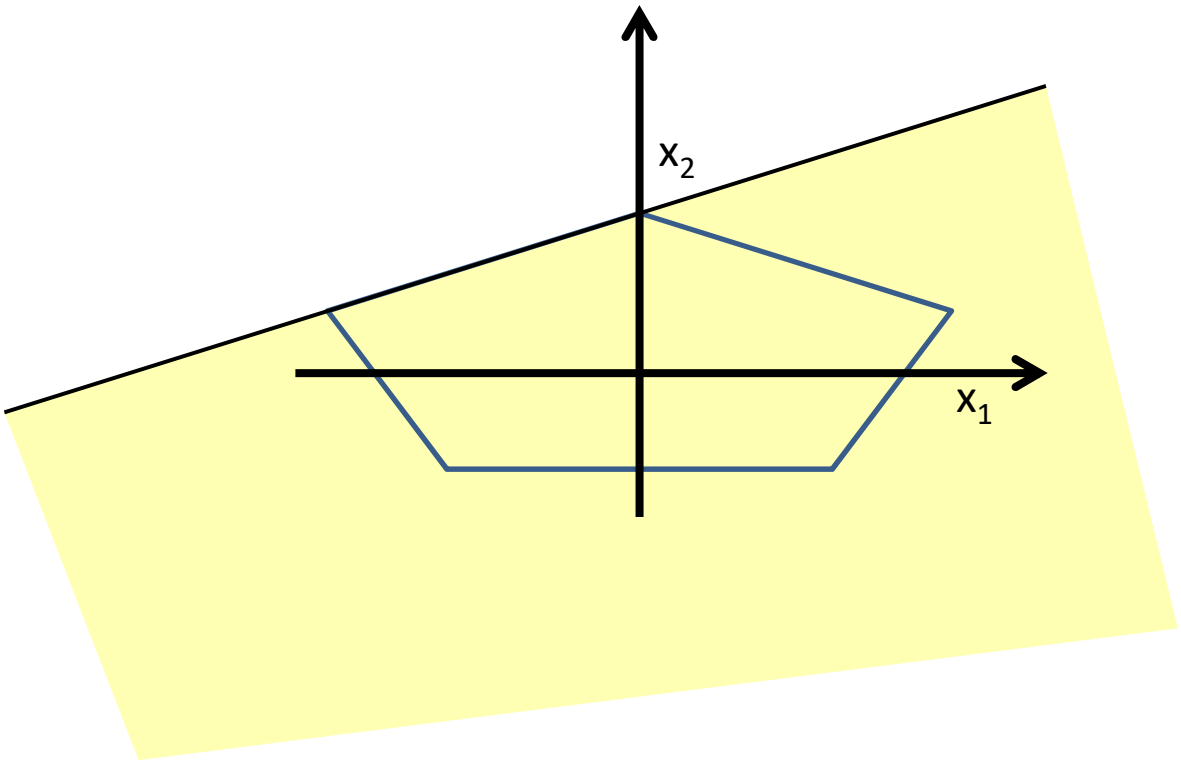

- The network must fire if the input is in the coloured area

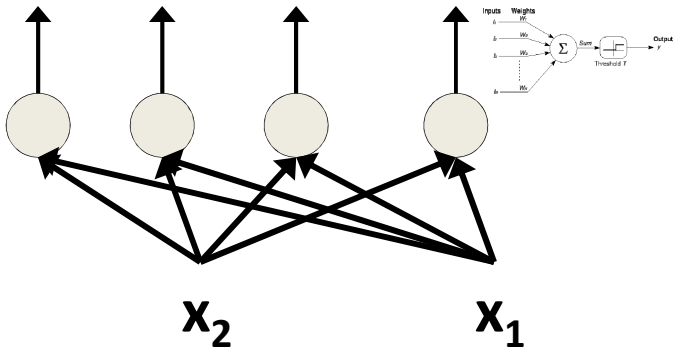
# Booleans over the reals




- The network must fire if the input is in the coloured area

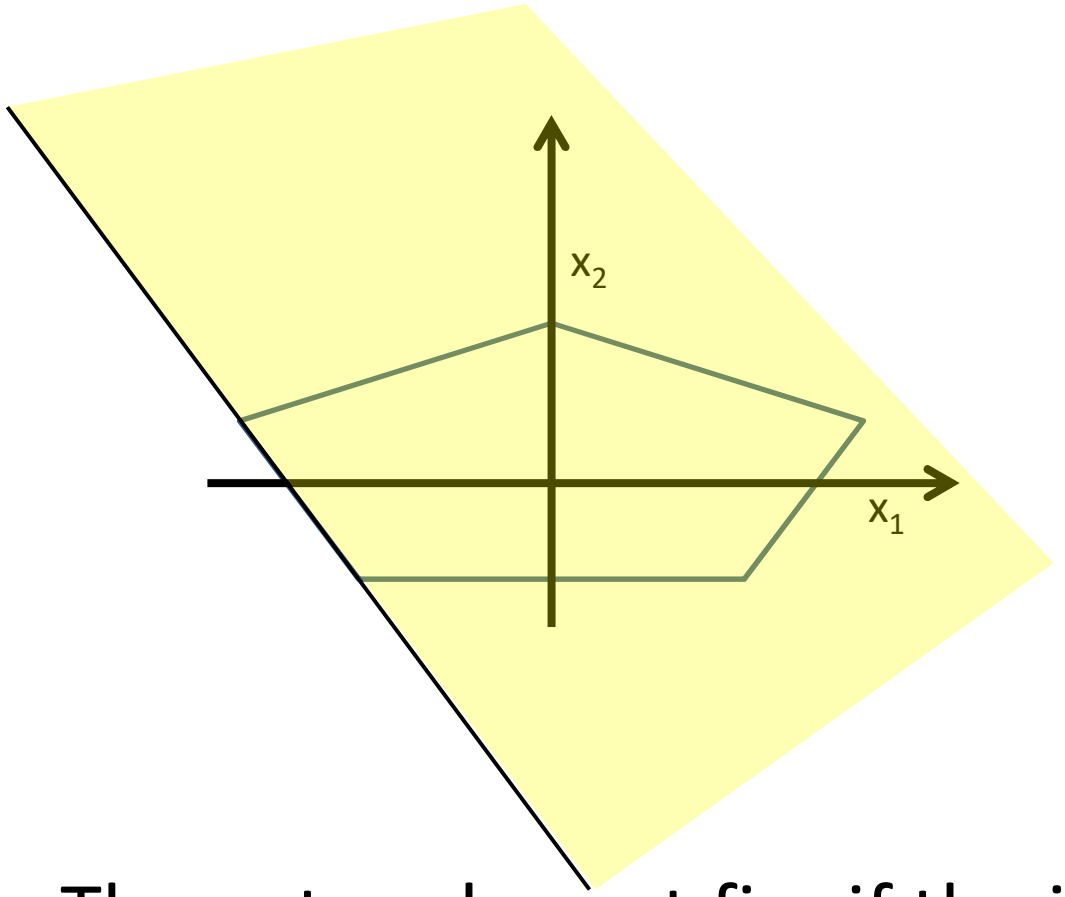
# Booleans over the reals

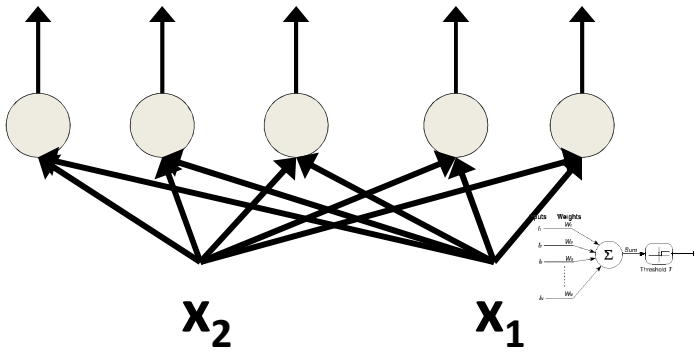



- The network must fire if the input is in the coloured area

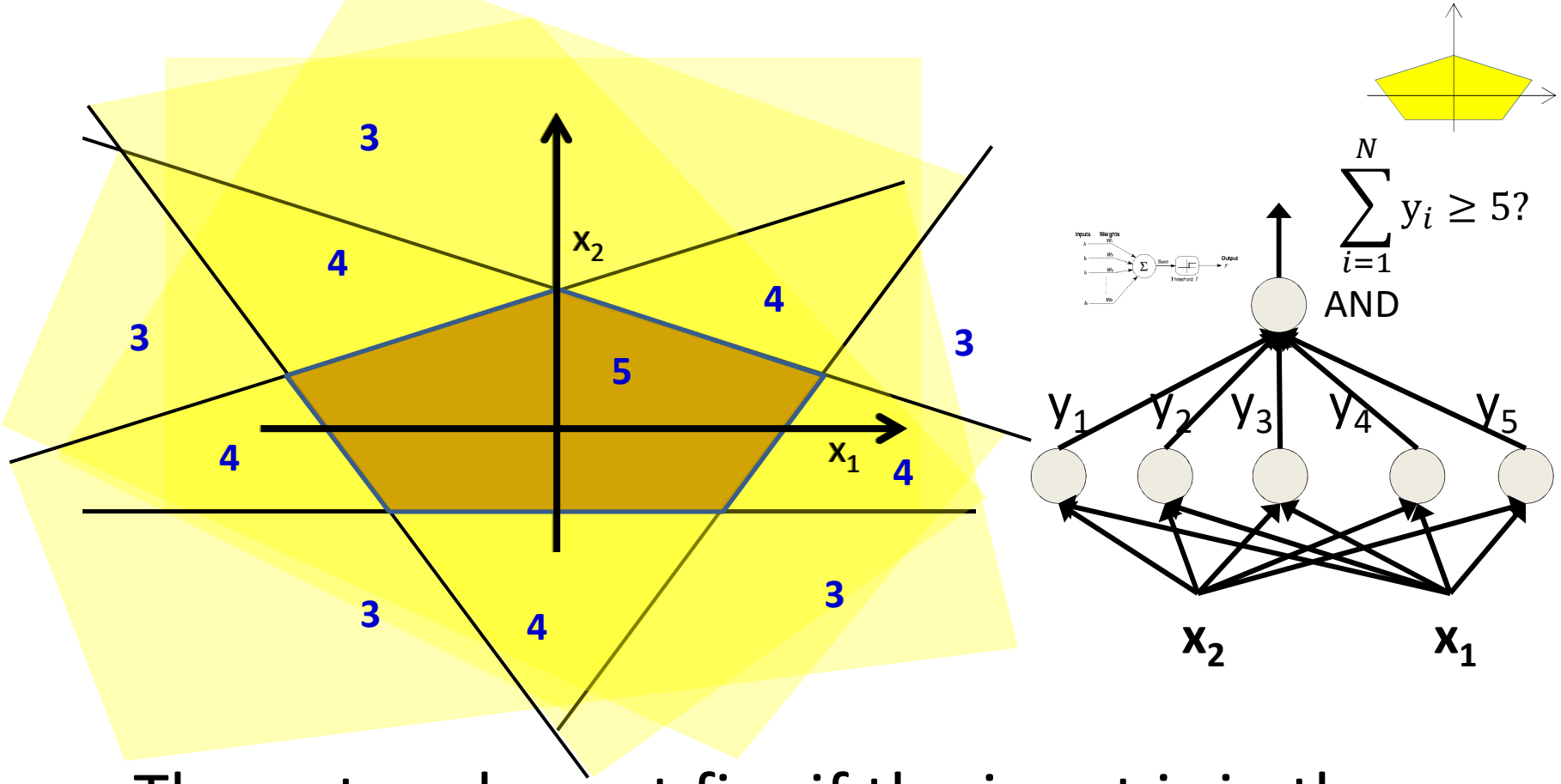
# Booleans over the reals



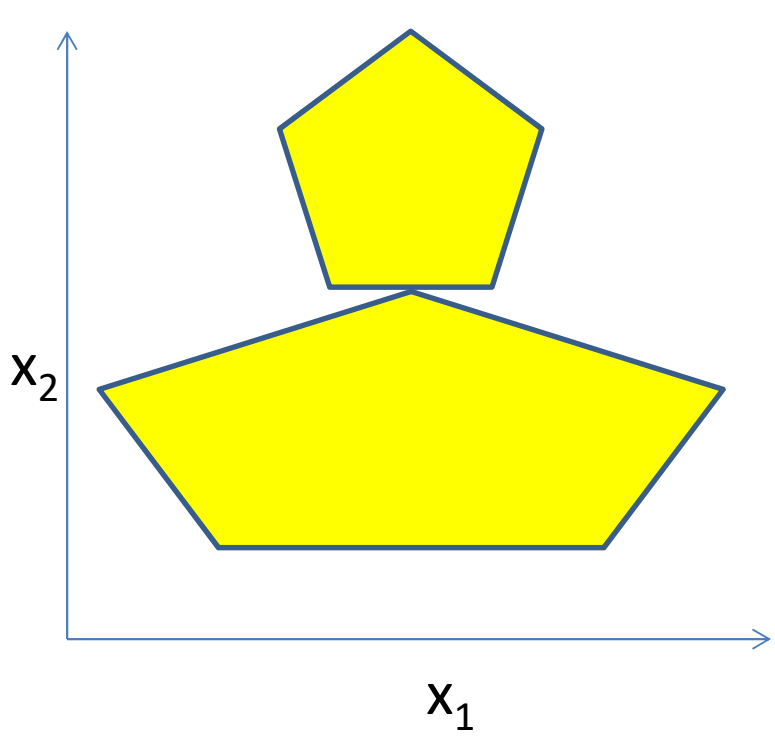

- The network must fire if the input is in the coloured area

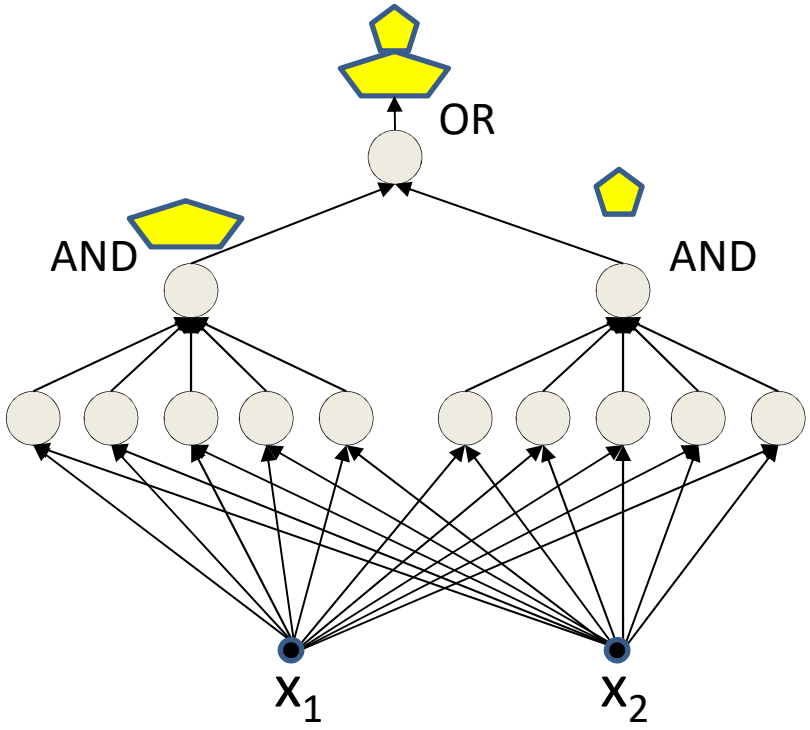
# Booleans over the reals




- The network must fire if the input is in the coloured area

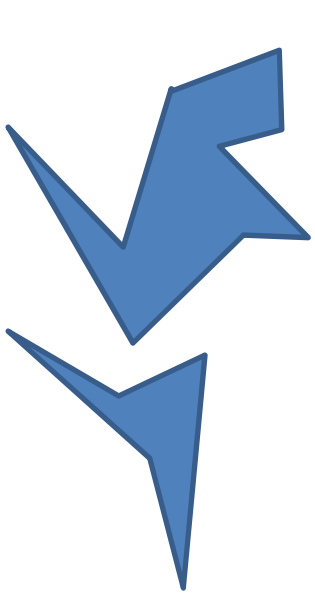
# More complex decision boundaries

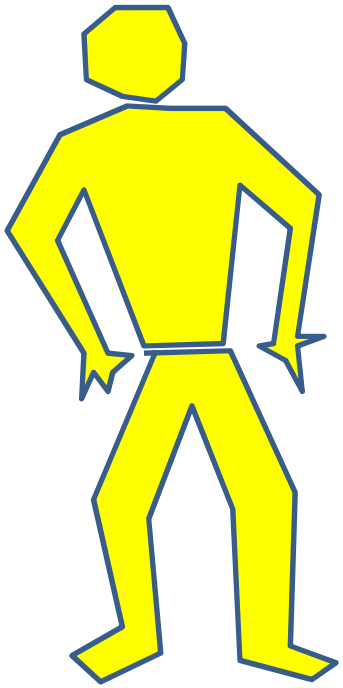


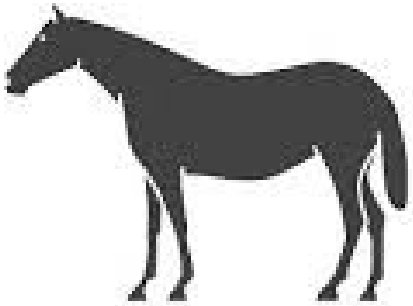




- Network to fire if the input is in the yellow area
  - "OR" two polygons
  - A third layer is required

# Complex decision boundaries

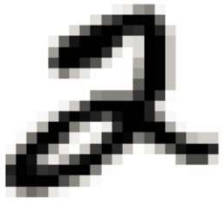




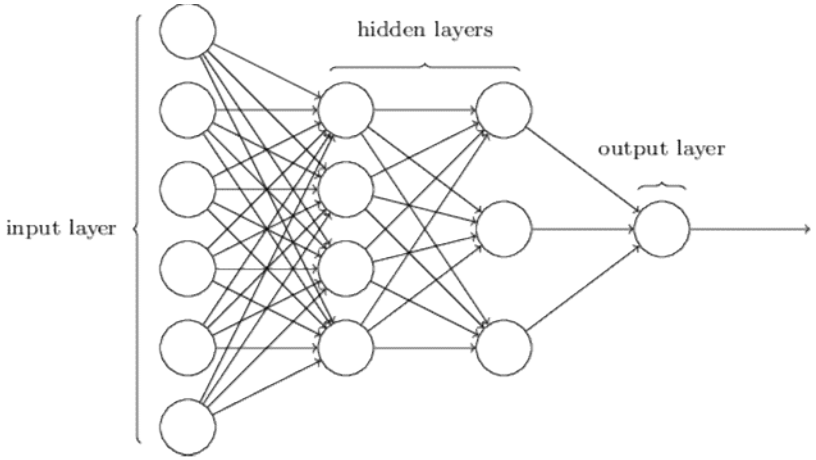


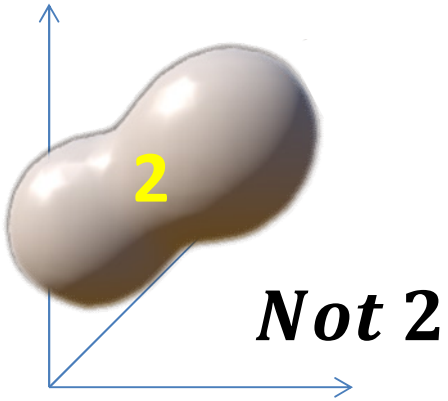
- Can compose very complex decision boundaries
  - How complex exactly? More on this in the next class

# Complex decision boundaries



784 dimensions
(MNIST)



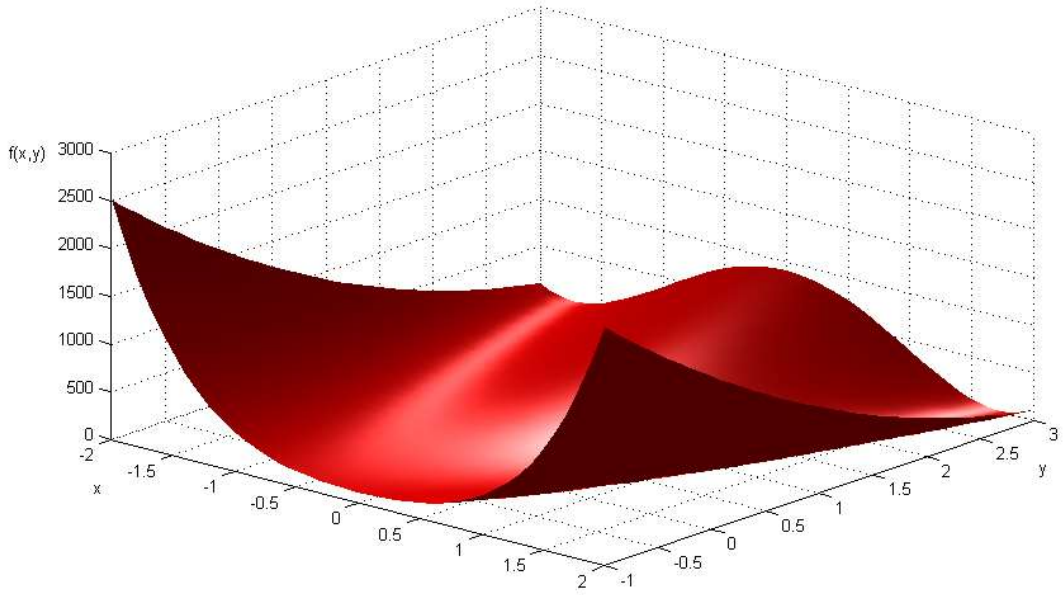

784 dimensions

- Classification problems: finding decision boundaries in high-dimensional space
  - Can be performed by an MLP
- MLPs can *classify* real-valued inputs

# Story so far

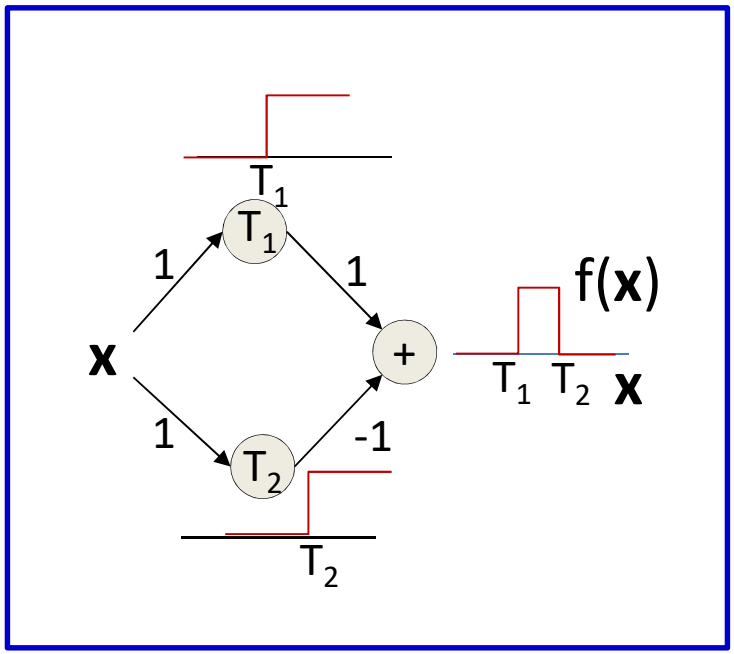
- **MLPs are connectionist computational models**
  - Individual perceptrons are computational equivalent of neurons
  - The MLP is a layered composition of many perceptrons
- **MLPs can model Boolean functions**
  - Individual perceptrons can act as Boolean gates
  - Networks of perceptrons are Boolean functions
- **MLPs are Boolean *machines***
  - They represent Boolean functions over linear boundaries
  - They can represent arbitrary decision boundaries
  - They can be used to *classify* data

# But what about continuous valued *outputs?*



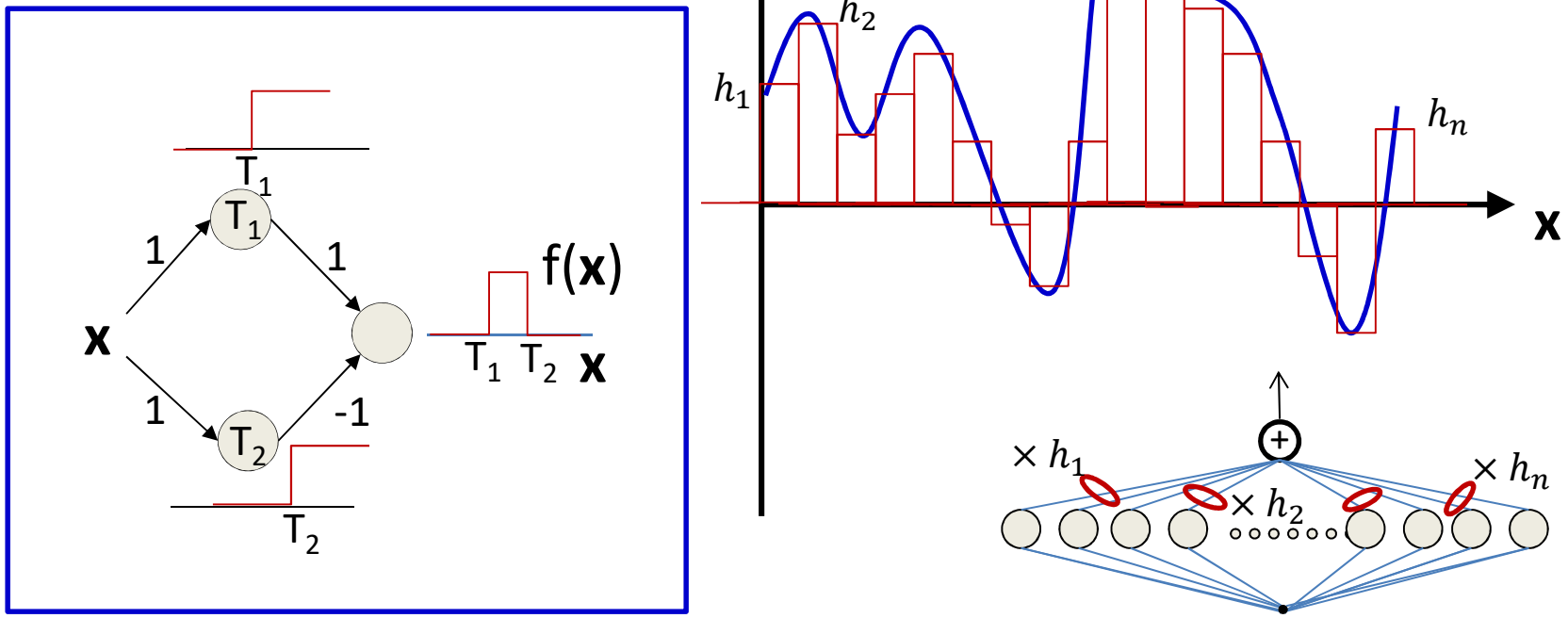

- Inputs may be real valued
- Can outputs be continuous-valued too

# MLP as a continuous-valued regression




- A simple 3-unit MLP with a "summing" output unit can generate a "square pulse" over an input
  - Output is 1 only if the input lies between $T_1$ and $T_2$
  - $T_1$ and $T_2$ can be arbitrarily specified

# MLP as a continuous-valued regression




- A simple 3-unit MLP can generate a "square pulse" over an input
- **An MLP with many units can model an arbitrary function over an input**
  - To arbitrary precision
    - Simply make the individual pulses narrower
- This generalizes to functions of any number of inputs (next class)

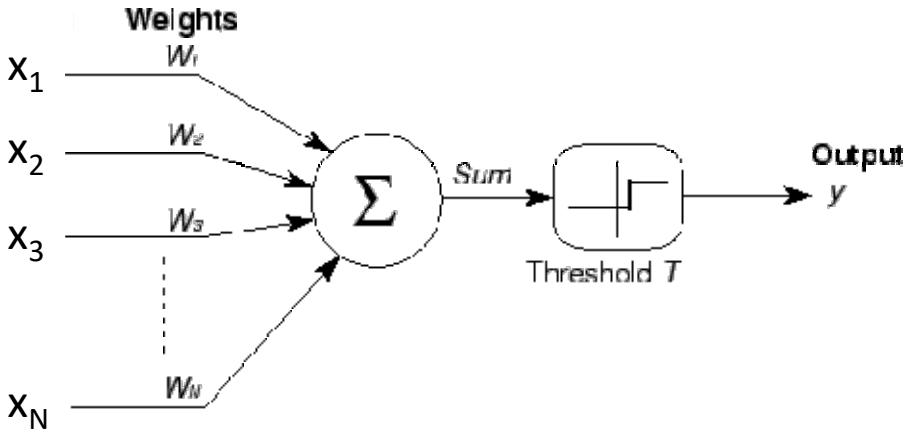
# Story so far

- **Multi-layer perceptrons are connectionist computational models**
- **MLPs are *classification engines***
  - They can identify classes in the data
  - Individual perceptrons are feature detectors
  - The network will fire if the combination of the detected basic features matches an "acceptable" pattern for a desired class of signal
- **MLP can also model continuous valued functions**

# So what does the perceptron really model?

- Is there a “semantic” interpretation?
  - Cognitive version: Is there an interpretation beyond the simple characterization as Boolean functions over sensory inputs?

# Lets look at the weights

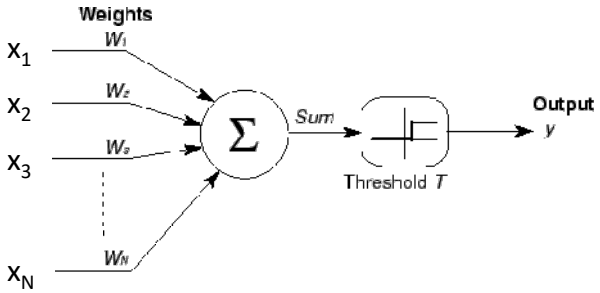



$$y = \begin{cases} 1 \; if \; \sum_i w_i x_i \geq T \\ 0 \; else \end{cases}$$

$$y = \begin{cases} 1 \; if \; \mathbf{x}^T \mathbf{w} \geq T \\ 0 \; else \end{cases}$$

- What do the *weights* tell us?
  - The neuron fires if the inner product between the weights and the inputs exceeds a threshold

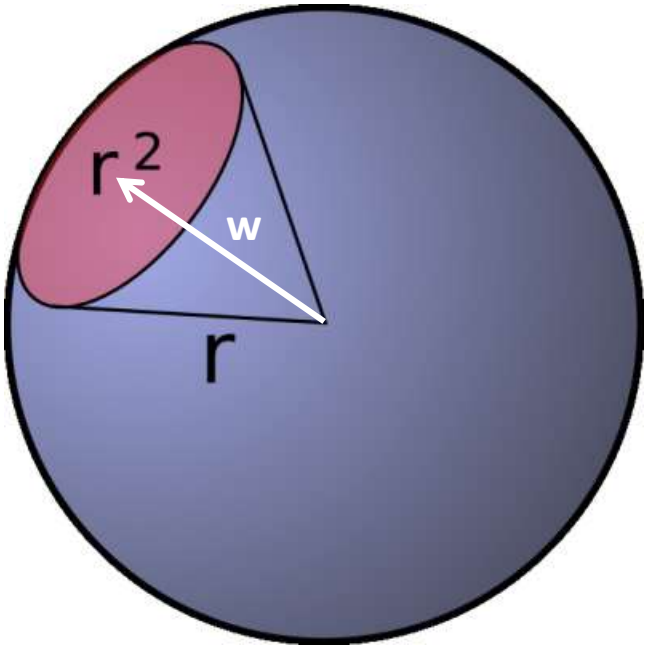
# The weight as a "template"




$$X^T W > T$$

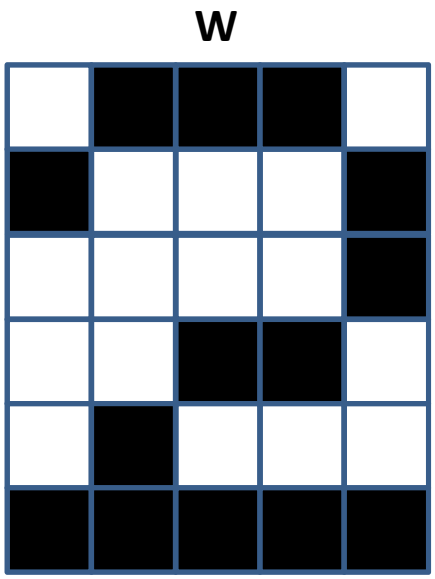
$$\cos\theta > \frac{T}{|X|}$$

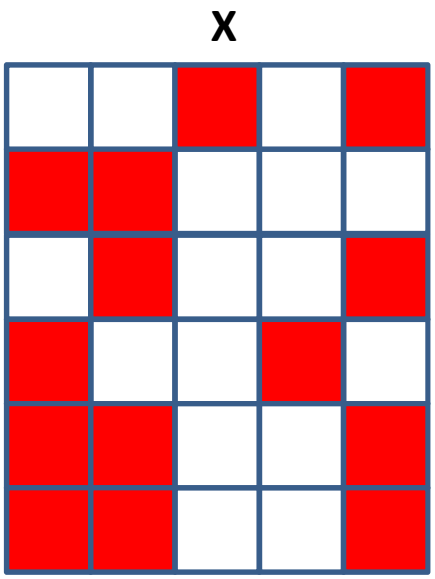
$$\theta < cos^{-1}\left(\frac{T}{|X|}\right)$$



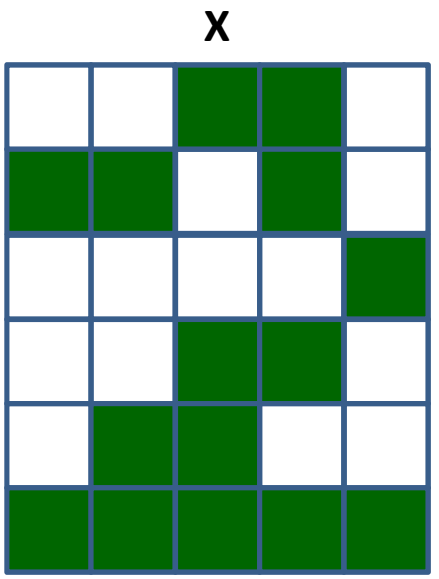

- The perceptron fires if the input is within a specified angle of the weight
- Neuron fires if the input vector is close enough to the weight vector.
  - If the input pattern matches the weight pattern closely enough

# The weight as a template




Correlation = 0.57

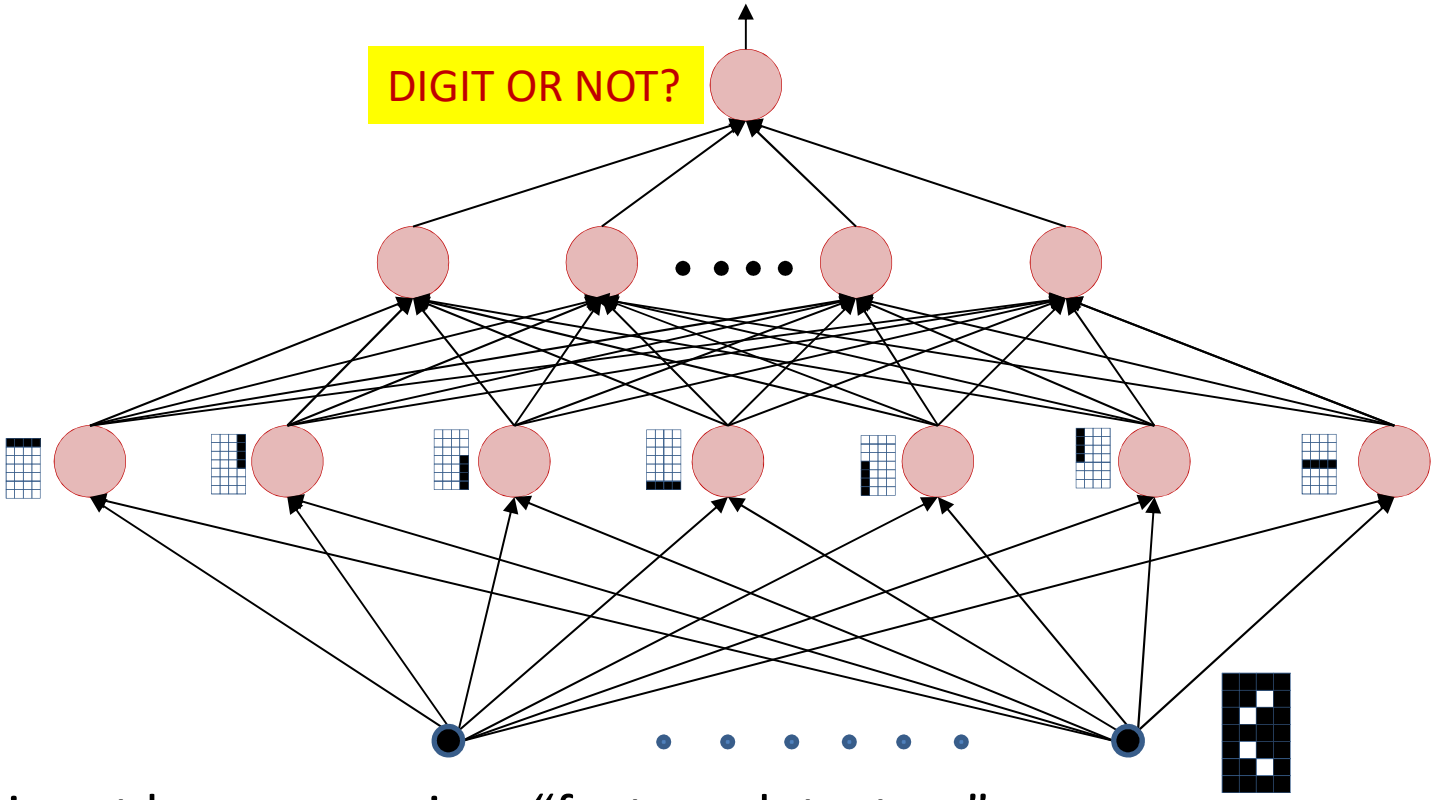



Correlation = 0.82

$$y = \begin{cases} 1 \; if \sum_i w_i x_i \geq T \\ 0 \; else \end{cases}$$

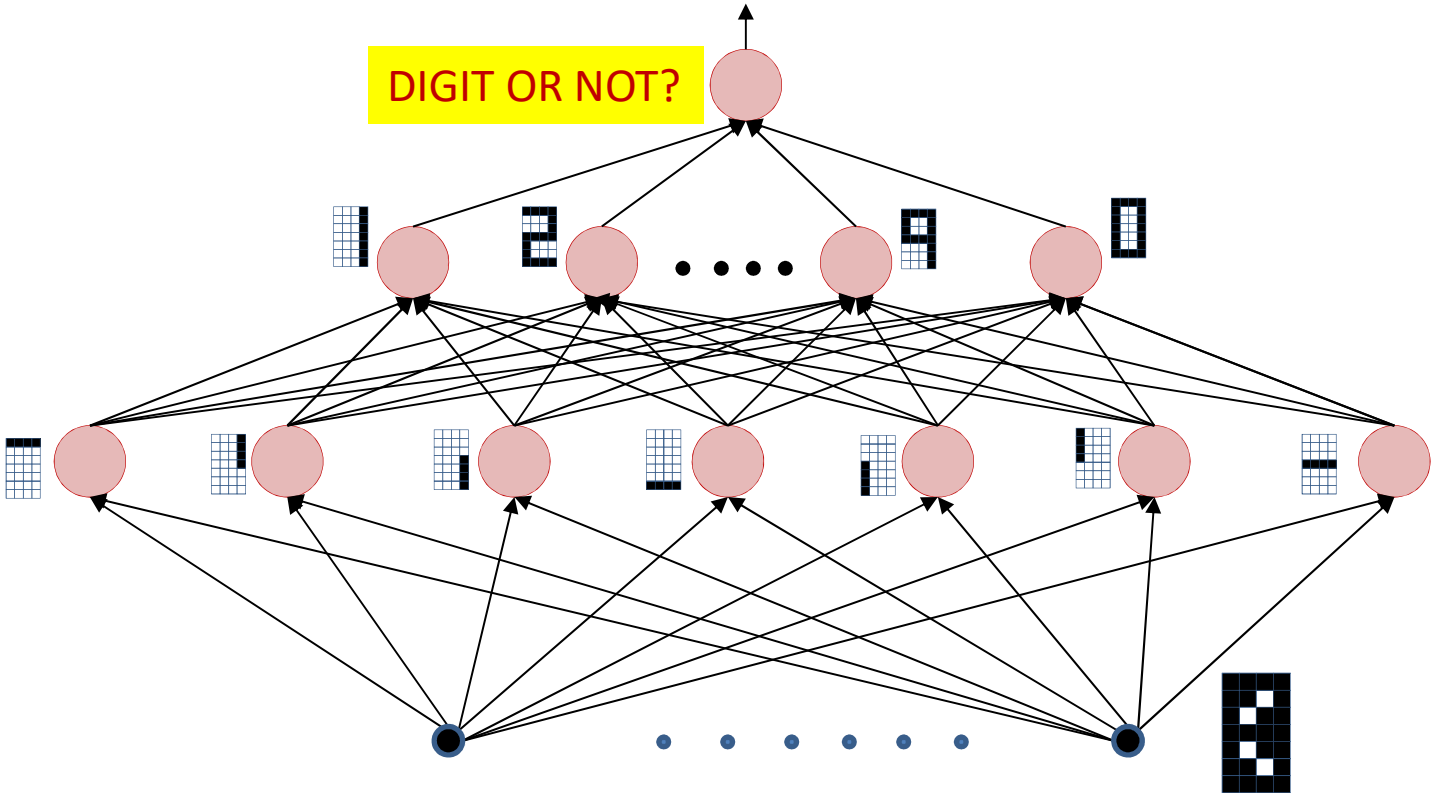
- If the *correlation* between the weight pattern and the inputs exceeds a threshold, fire
- The perceptron is a *correlation filter!*

# The MLP as a Boolean function over feature detectors




- The input layer comprises “feature detectors”
  - Detect if certain patterns have occurred in the input
- The network is a Boolean function over the feature detectors
- I.e. it is important for the *first* layer to capture relevant patterns

# The MLP as a cascade of feature detectors



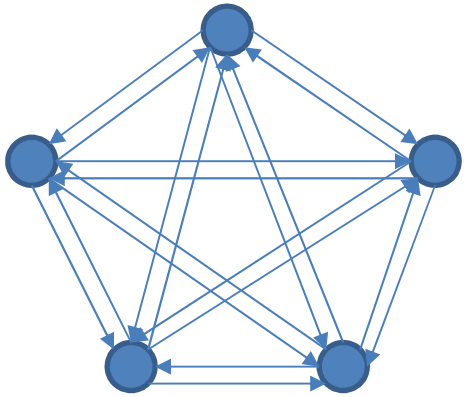

- The network is a cascade of feature detectors
  - Higher level neurons compose complex templates from features represented by lower-level neurons

# Story so far

- **Multi-layer perceptrons are connectionist computational models**
- **MLPs are Boolean *machines***
  - They can model Boolean functions
  - They can represent arbitrary decision boundaries over real inputs
- **MLPs can approximate continuous valued functions**
- **Perceptrons are *correlation filters***
  - They detect patterns in the input

# Other things MLPs can do

- Model memory
  - Loopy networks can “remember” patterns
    - Proposed by Lawrence Kubie in 1930, as a model for memory in the CNS



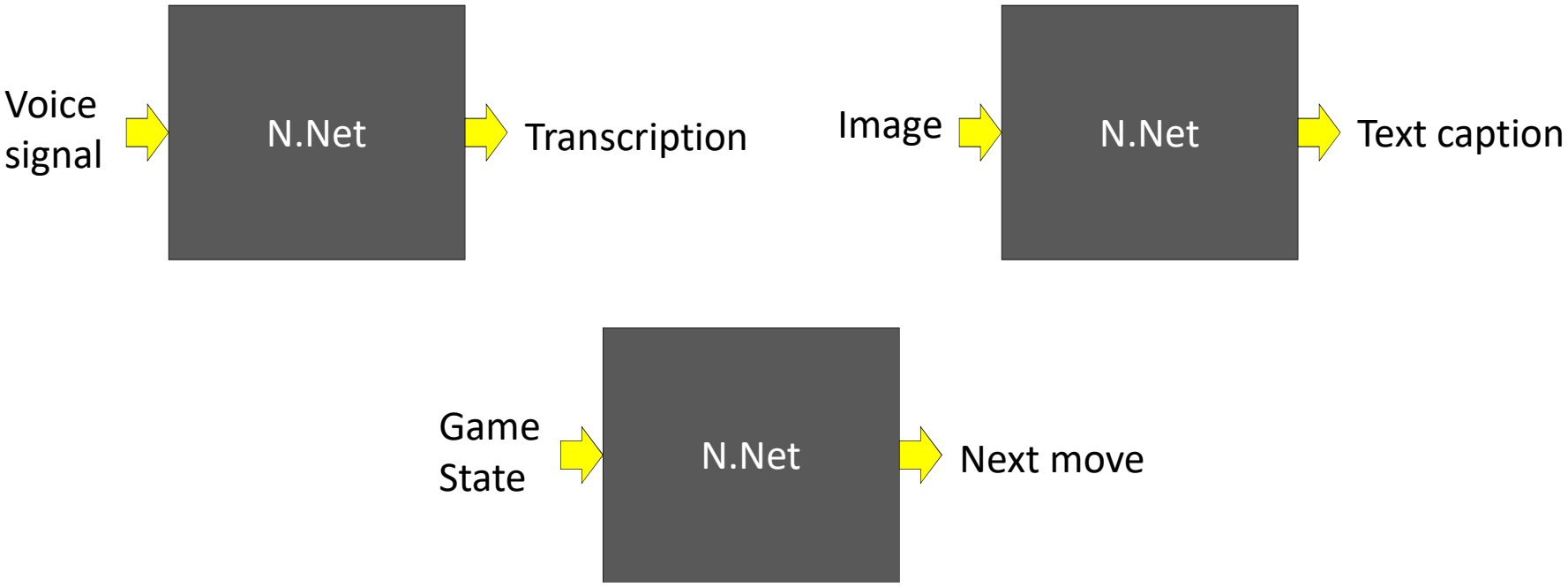
- Represent probability distributions
  - Over integer, real and complex-valued domains
  - MLPs can model both *a posteriori* and *a priori* distributions of data
    - A posteriori conditioned on other variables

- They can rub their stomachs and pat their heads at the same time..

# NNets in AI

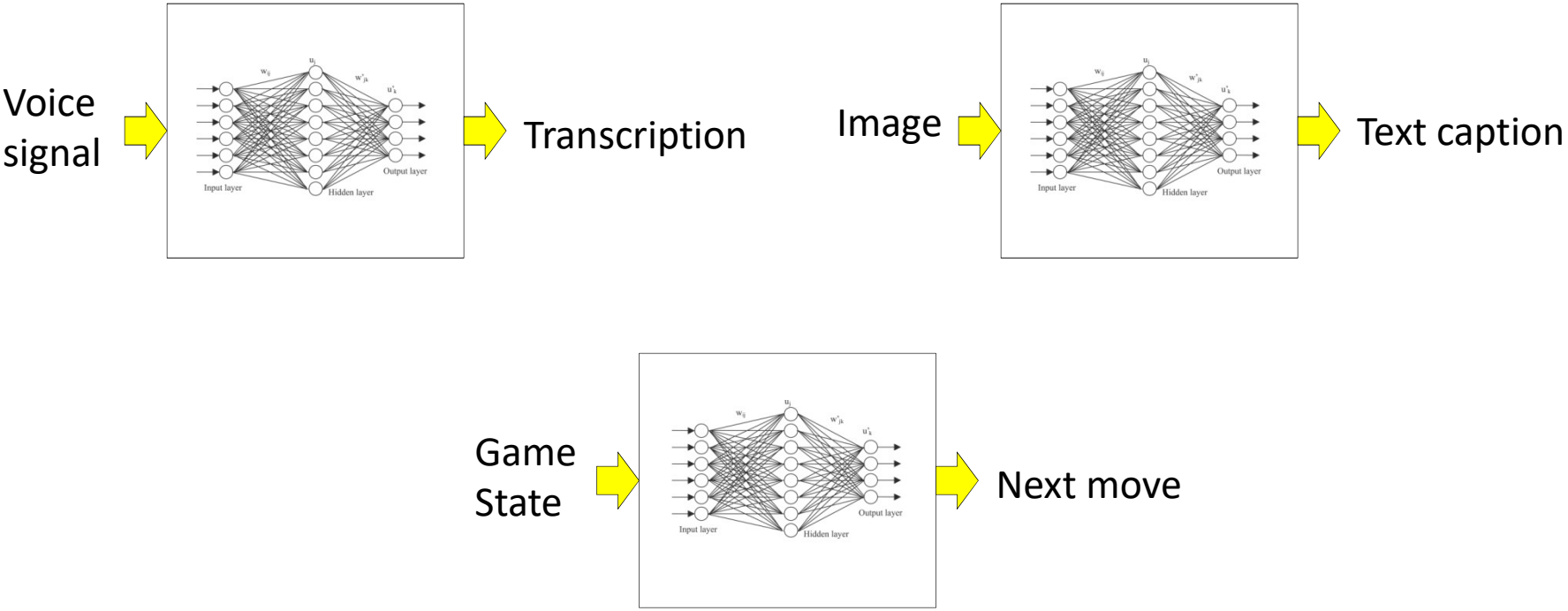
- The network is a function
  - Given an input, it computes the function layer wise to predict an output

# These tasks are *functions*



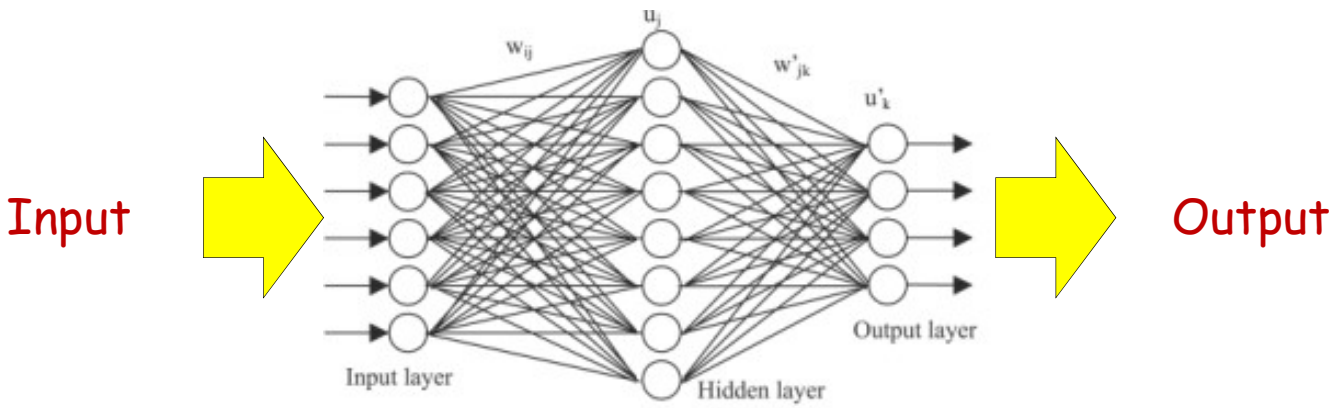

- Each of these boxes is actually a function
  - E.g f: Image → Caption

# These tasks are *functions*




- Each box is actually a function
  - E.g f: Image → Caption
  - It can be approximated by a neural network

# The network as a function




- Inputs are numeric vectors
  - Numeric representation of input, e.g. audio, image, game state, etc.
- Outputs are numeric scalars or vectors
  - Numeric “encoding” of output from which actual output can be derived
  - E.g. a score, which can be compared to a threshold to decide if the input is a face or not
  - Output may be multi-dimensional, if task requires it

# Story so far

- Multi-layer perceptrons are connectionist computational models
- MLPs are *classification engines*
- MLP can also model continuous valued functions
- Interesting AI tasks are functions that can be modelled by the network

# Next Up

- More on neural networks as universal approximators
  - And the issue of depth in networks