Artificial Intelligence foundations
DD1390 Guest Lecture

Magnus Boman (mab@kth.se)

April 7, 2020

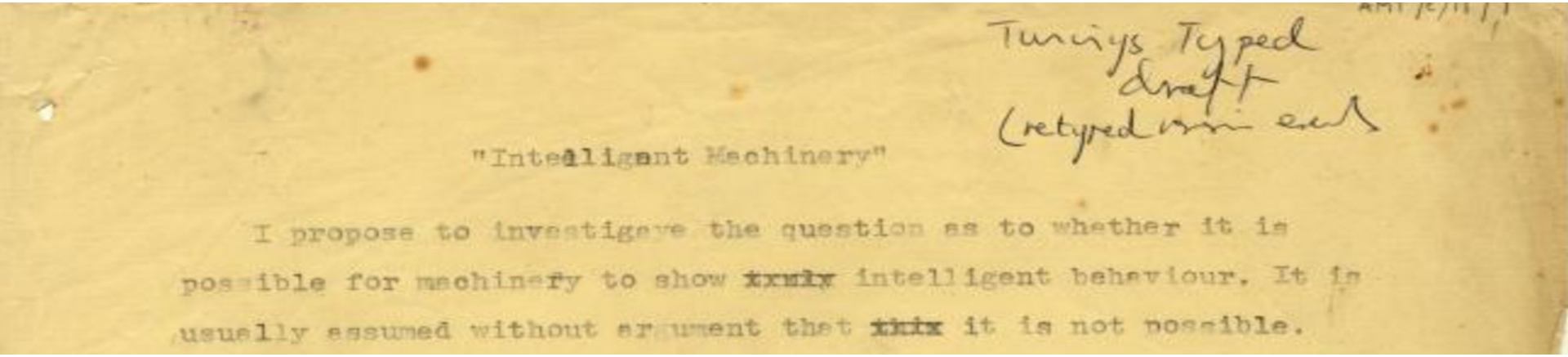
Magnus Boman

- Professor in Intelligent Software Services at KTH since 2003
- One day per week at KI in the AI@KI project
- Affiliated with KI, King's College London, and University College London
- Fellow at the Royal Society of Medicine, London
- Associate editor of Eurosurveillance

One possible historical narrative

- 1930s (programming begins, lambda calculus style)
- 1940s (operations research + first hardware)
- 1950s (Turing machines, von Neumann, AI)
- 1960s (math, combinatorial optimization, …)
- 1970s (problem solving with computers)
- 1980s (programming gets scalable, databases)
- 1990s (supercomputing, distributed knowledge bases)
- 2000s (cloud, DSLs, scalable OOP,…)
- 2010s (qubits, reservoirs, scalable data science,…)
- 2020s (real-time stream proc, unsupervised transfer learning, …)

Alan Turing (1948 NPL paper, digital archive)

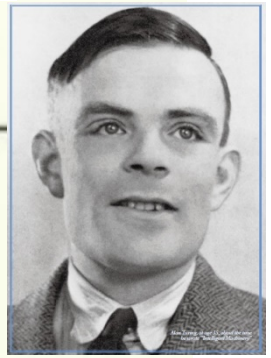


- A ‘schoolboy essay’ with ‘smudgy’ appearance, according to Charles Darwin (director, NPL)
- In fact, *the* manifesto of AI, with a full review of connectionist models, including ‘genetical search’

Alan Turing (1948 paper published in 1968)

Intelligent Machinery

A. M. Turing
[1912—1954]



Abstract

The possible ways in which machinery might be made to show intelligent behaviour are discussed. The analogy with the human brain is used as a guiding principle. It is pointed out that the potentialities of the human intelligence can only be realized if suitable education is provided. The investigation mainly centres round an analogous teaching process applied to machines. The idea of an unorganized machine is defined, and it is suggested that the infant human cortex is of this nature. Simple examples of such machines are given, and their education by means of rewards and punishments is discussed. In one case the education process is carried through until the organization is similar to that of an ACE.

Connectionist machines or Artificial Neural Networks (McCulloch+Pitts 1943)

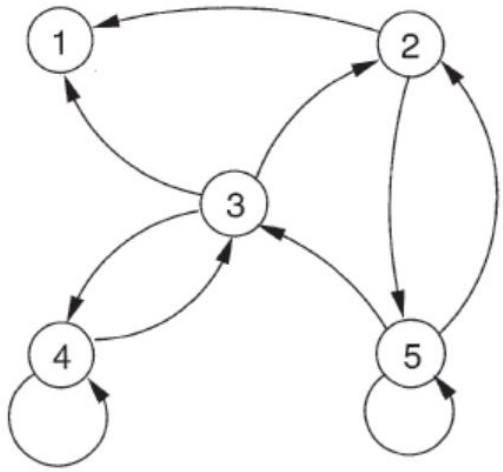
- Implement logical expressions as nets of simplified neurons (in which the axon of one is connected to the soma of another via a synapse)
- Each neuron is binary and has a finite threshold
- Each synapse is either excitory or inhibitory and causes a finite delay of one cycle
- Networks can be constructed with multiple synapses between any pair

Unorganized (connectionist) machines

- A-type
  Randomly connected neurons (NAND gates), each with two inputs, synchronized by a global clock signal for each moment, in which a unit is in state 0 or state 1
- B-type
  Like A, but with modifiable interconnections (switch), allowing for organizing the machine by enabling/ disabling (reinforcement)

A-type unorganized machines

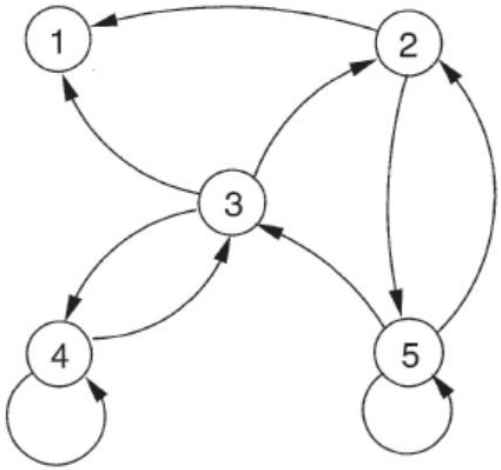
- The simplest model of a nervous system with a random arrangement of neurons
- Digital
- Deterministic
- Non-modifiable
- Finite memory
- Output eventually periodic
- Random Boolean Network or
  Random Binary Recurrent Network



Example state sequence of a 5-state machine

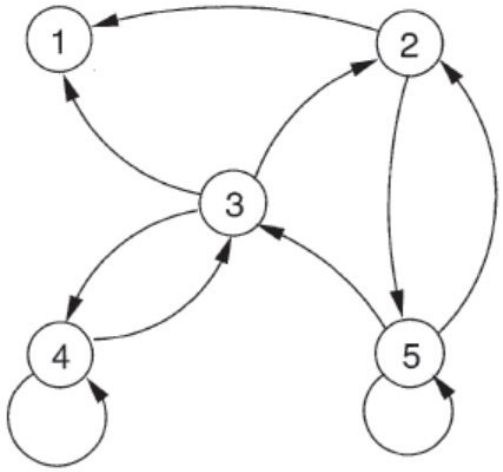
- Compute the state sequence from time *t* five steps ahead, given the initial configuration:

```
1 = 1
2 = 1
3 = 0
4 = 0
5 = 1
```



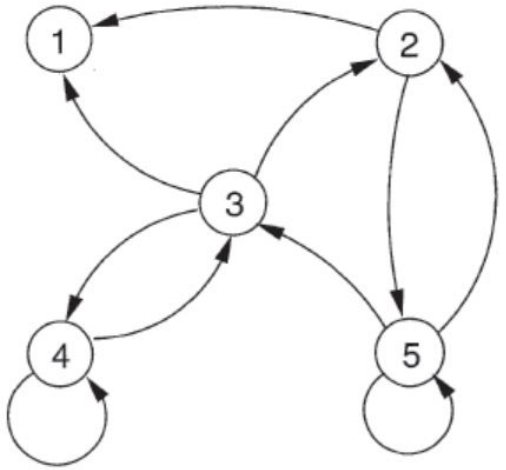
Example state sequence of a 5-state machine

	t	t+1	t+2	t+3	t+4	t+5
1 =	1					
2 =	1					
3 =	0					
4 =	0					
5 =	1					



Example state sequence of a 5-state machine

	t	t+1	t+2	t+3	t+4	t+5
1 =	**1**	1	0	0	1	0
2 =	**1**	1	1	0	1	0
3 =	**0**	1	1	1	1	1
4 =	**0**	1	0	1	0	1
5 =	**1**	0	1	0	1	0



Single-layer simple machines impress

Larry Bull, Julian Holley, Ben De Lacy Costello & Andrew Adamatzky (2013)
Toward Turing's A-type Unorganised Machines in an Unconventional Substrate:
a Dynamic Representation in Compartmentalised Excitable Chemical Media.
In Dodig-Crnkovic & Giovagnoli (Eds) Computing Nature. Springer.

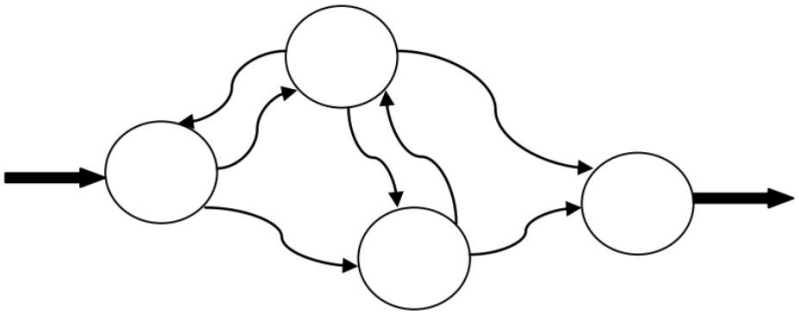


Fig. 1. Simple example A-type unorganised machine consisting of four two-input NAND gate nodes (N=4), with one input (node 1) and one output (node 4) as indicated by the bold arrows.

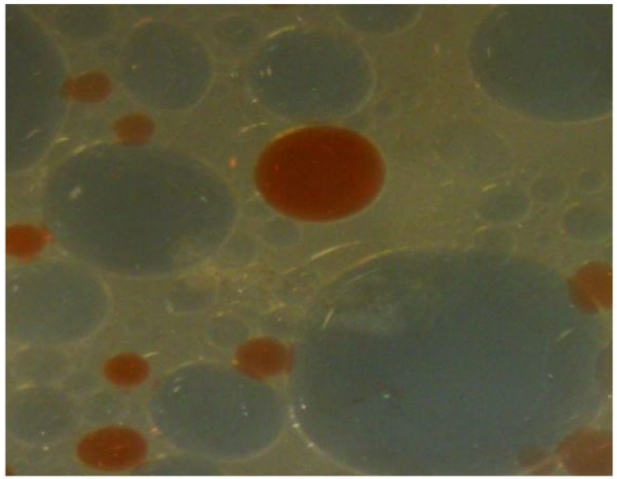


Fig. 3. Showing the BZ droplet vesicles.

B-type unorganized machines

- Any number of neurons, connected in any pattern
- Each neuron-neuron connection must pass through a modifier
- Each modifier device has two training fibers, a pulse to one fiber sets it to Pass mode, a pulse to the other sets it to Interrupt mode

  Pass: $0 \rightarrow 0, 1 \rightarrow 1$

  Interrupt: $0 \rightarrow 1, 1 \rightarrow 1$

Modularity: B-type is A-type with A-type machines replacing connections (Teuscher)

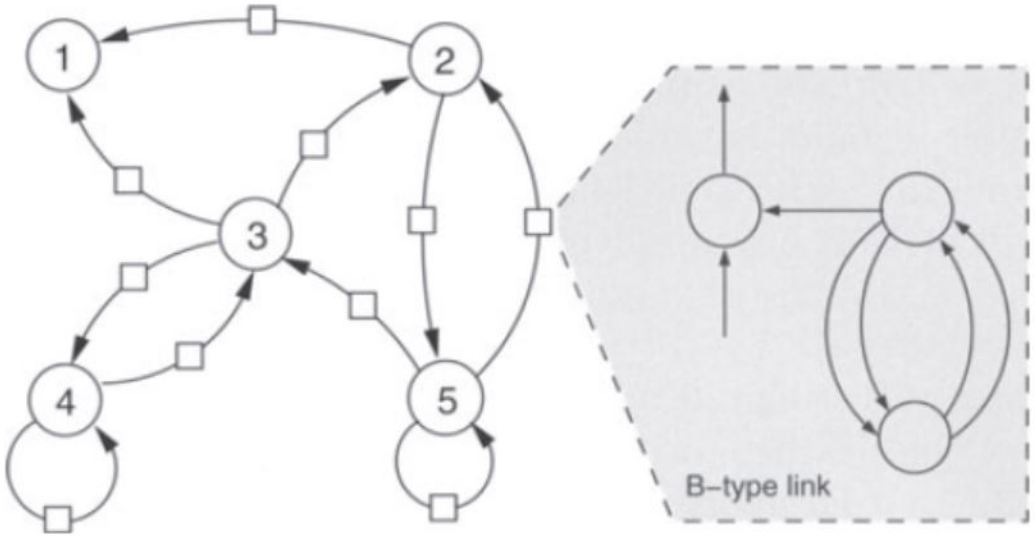


Fig. 8. Example of a B-type unorganized machine built up from five units. Each B-type link is itself a small A-type machine and the entire machine might also be considered as a specifically organized A-type machine

Random Boolean Networks (Random Binary Recurrent Network)

- The most organized behaviour in RBNs occur when each node receives input from two other nodes on average (Kauffman 1993)
- Turing's unorganized machines are a subset of RBNs in which K=2 and the Boolean function is a fixed NAND function for each node

Unorganized machines = Artificial Neural Networks

- P-type (trainable)
- Not connectionist, but a tape-less Turing machine that has two additional inputs, for Pain and Pleasure
- By means of an external educator, stimuli via the two P-inputs complete the internal table of the state transition function

Genetical search (Turing 1948)

"There is the genetical or evolutionary search by which a combination of genes is looked for, the criterion being survival value. The remarkable success of this search confirms to some extent the idea that intellectual activity consists mainly of various kinds of search."

Adaptation

- is a walk in the parameter space of a dynamic system that promotes homeostasis
- is related to stability and survival
- can be modelled by basins of attraction and jumping states in a complex system
- in Turing's unorganised machines modifies the switches between the neurons

Adaptation
Ross Ashby: Design for a Brain (2nd ed, 1960)

5/8. We can now recognise that '*adaptive*' *behaviour is equivalent to the behaviour of a stable system, the region of the stability being the region of the phase-space in which all the essential variables lie within their normal limits.*

The view is not new (though it can now be stated with more precision):

> 'Every phase of activity in a living being must be not only a necessary sequence of some antecedent change in its environment, but must be so adapted to this change as to tend to its neutralisation, and so to the survival of the organism. . . . It must also apply to *all* the relations of living beings. It must therefore be the guiding principle, not only in physiology . . . but also in the other branches of biology which treat of the relations of the living animal to its environment and of the factors determining its survival in the struggle for existence.'
>
> (Starling.)
>
> 'In an open system, such as our bodies represent, compounded of unstable material and subjected continuously to disturbing conditions, constancy is in itself evidence that agencies are acting or ready to act, to maintain this constancy.'
>
> (Cannon.)
>
> 'Every material system can exist as an entity only so long as its internal forces, attraction, cohesion, etc., balance the

> external forces acting upon it. This is true for an ordinary stone just as much as for the most complex substances; and its truth should be recognised also for the animal organism. Being a definite circumscribed material system, it can only continue to exist so long as it is in continuous equilibrium with the forces external to it: so soon as this equilibrium is seriously disturbed the organism will cease to exist as the entity it was.'
>
> (Pavlov.)

McDougall never used the concept of 'stability' explicitly, but when describing the type of behaviour which he considered to be most characteristic of the living organism, he wrote:

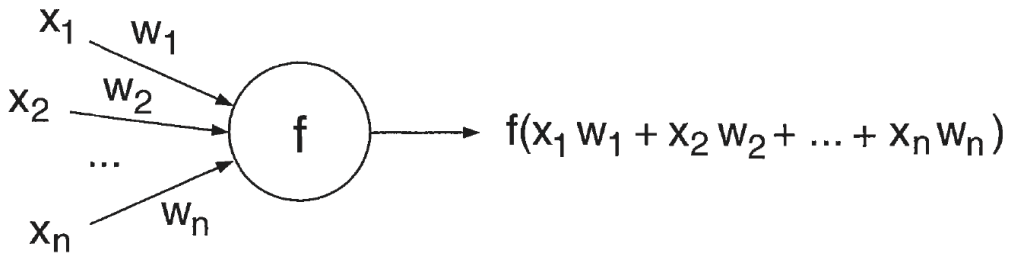
> 'Take a billard ball from the pocket and place it upon the table. It remains at rest, and would continue to remain so for an indefinitely long time, if no forces were applied to it. Push it in any direction, and its movement in that direction persists until its momentum is exhausted, or until it is deflected by the resistance of the cushion and follows a new path mechanically determined. . . . Now contrast with this an instance of behaviour. Take a timid animal such as a guinea-pig from its hole or nest, and put it upon the grass plot. Instead of remaining at rest, it runs back to its hole; push it in any other direction, and, as soon as you withdraw your hand, it turns back towards its hole; place any obstacle in its way, and it seeks to circumvent or surmount it, restlessly persisting until it achieves its end or until its energy is exhausted.'

He could hardly have chosen an example showing more clearly the features of stability.

Evolutionary Artificial Neural Networks

Evolutionary ANNs use evolution as its adaptation mechanism, at three levels [Yao]:

1. Connection weights (allows for training)
2. Architectures (allows for self-adaptation)
3. Learning rules (allows for learning machines)



Neural network design by evolution

- Each ‘genome’ (string, chromosome) will code network structure and/or network weights
- A ‘genotype’ is an encoded network structure
- A ‘phenotype’ is a structure emerging from the interpretation of a genotype
- The ‘fitness’ of a network is the number of correctly classified new patterns

Logical Computing Machines (Turing Machines)

- Invented by Turing and published in 1937
- Resting on the Church-Turing thesis
- Introduced the concept of sub-routine into programming

Universal Turing machines

- Generalise Turing machines to machine schemata (‘main’)
- A single machine will suffice
- Engineering is replaced by “the office work of programming the universal machine”
- Can do anything described by a rule of thumb (heuristics) or something purely mechanical

Paper machines

- An abstract computer can be described on paper: a set of procedural rules can produce the same output as a TM
- “A man provided with paper, pencil, and rubber, and subject to strict discipline, is in effect a universal machine”

Machines without bodies

One way of setting about our task of building a 'thinking machine' would be to take a man as a whole and to try to replace all the parts of him by machinery. He would include television cameras, microphones, loudspeakers, wheels and 'handling servo-mechanisms' as well as some sort of 'electronic brain'. This would be a tremendous undertaking of course. The object, if produced by present techniques, would be of immense size, even if the 'brain' part were stationary and controlled the body from a distance. In order that the machine should have a chance of finding things out for itself it should be allowed to roam the countryside, and the danger to the ordinary citizen would be serious. Moreover even when the facilities mentioned above were provided, the creature would still have no contact with food, sex, sport and many other things of interest to the human being. Thus although this method is probably the 'sure' way of producing a thinking machine it seems to be altogether too slow and impracticable.

"Turing is going to infest the countryside with a robot which will live on twigs and scrap iron!"

Reasons for scepticism (Turing)

a) Humankind is the supreme intellectual entity
b) Religion ("Promethean irreverence")
c) Lack of examples of machine intelligence
d) Machines are bound by incompleteness theorems
e) Machines can only reflect the intelligence of their creator

Machine education (Turing 1951 talk)

- The machine occupies itself with playing GO (!)
- A ‘highly competent schoolmaster’ is its educator
- The schoolmaster is denied detailed information on the inner workings of the machine
- A ‘mechanic’ can do machine resets, but no education

Machine education components (1951)

- Memory (chronological log)
- ‘Indexes of experience’ (event-based log)
- Special features observed in the indexes already used
- Experience of outcome (positive and negative weights)
- Crude rules of thumb for outcome classification (supervised), over time replace by sophisticated rules (unsupervised)
- Random noise (obtainable from pseudo-random tape contents)

Unorganized machines are random

- Rules of procedure invoking random instructions is described as ‘(apparently) partially random’ (non-deterministic)
- A TM without such calls is ‘determined’ (deterministic)
- Machines made up in an unsystematic way are ‘unorganized’
- Interference via input, with the purpose of setting up for useful tasks, is ‘organizing the machine’ via modification

Random machines

- The cortex of a human infant is an unorganized machine that requires discipline and initiative to become intelligent
- Experiments with unorganized machines that try to organize them amount to modifying them into universal machines
- A ‘P-type unorganized machine’ is an incomplete TM, without a tape, but with memory and ‘pleasure-pain arrangements’

Machines learning via stimulus-response

- A machine receives stimuli via its sensors or receptor organs
- A machine acts or performs actions via its effectors
- That an action is a response to a stimulus means they co-occur
- Since both stimulus and response may be complicated, their relation is complicated, even in this simplest case

Time quantization in machines

- That an action is a response to a stimulus co-occurring at a particular time does not increase expressive power
- Neuron-firing is dichotomous, similar to biological systems, meaning that time is quantized
- Simple learning machines are (digital) automata, whereas some organic processes are analogue, like the human respiratory system (nervous response to blood CO_2 levels)
- The B-type switch by which an outsider can 'organize' them allows for learning, via reinforcement of successful links

Reinforcement via supervision

- Learning algorithms that are unsupervised may use

  - Reinforcement learning
  Weights updated only after I/O can be inspected and judged, and weight updating can be done with the input vector only
  - Error correction
  Magnitude of error and the input vector determine the correction of weights

Expressive power of simple machines

- A McCulloch-Pitts nerve net without circles can be represented in propositional logic
- Any McCulloch-Pitts nerve net is a finite automaton (Chomsky type-3, recognises regular languages), with more expressive power than A- and B-type, since they can have any number of inputs and have inhibitory/ exhibitory synapses
- There are simple neuronal events that cannot be represented by any nerve net

Expressive power of less simple machines

- A perceptron model with a linear transfer function is equivalent to a possibly multiple or multivariate linear regression model [Weisberg 1985; Myers 1986]
- A perceptron model with a logistic transfer function is a logistic regression model [Hosmer and Lemeshow 1989]
- A perceptron model with a threshold transfer function is a linear discriminant function [Hand 1981; McLachlan 1992; Weiss and Kulikowski 1991]

G. Hinton *et al.*, updated by N. Intrator, May 2007

Perfect neural networks

- A first proviso is that neurons never fire in error
- The theory of ANNs rests on this proviso
- The representation of an event can therefore be described by the firing of a single neuron
- A second proviso is that when designing learning machines, an *ad infinitum* behaviour is provided for: machines will not die

On achieving machine intelligence (1951)

- The machines will not die
- Religious tolerance needs to advance
- Fear of losing employment - “There would be great opposition from the intellectuals who were afraid of being put out of a job”
- Learning from machines - “...it seems probable that once the machine thinking method has started, it would not take long to outstrip our feeble powers”
- LM2LM - “...they would be able to converse with each other to sharpen their wits.”

The Hebbian rule

"Stored information takes the form of new connections, or transmission channels in the nervous system (or the creation of conditions which are functionally equivalent to new connections)"

F Rosenblatt (1958). The Perceptron. *Psychological Review* 65(6):399

Farley and Clark (1954) simulated the first ANN, although Rosenblatt built his Perceptron (1956) on the 'connection strength' of Hebb

Connectionism using the Hebbian rule

A mass of neurons can learn if their connection strengths change according to some rule

D. Hebb. The Organization of Behavior. John Wiley, New York, 1949

Optimum performance of learning machines (Gamba 1961)

- Learning is inductive rather than deductive
- Machines learn from examples
- Learning has an upper limit in that it cannot outperform an informed mathematician equipped with probability theory
- Random connections (as employed in the Perceptron) are useful for learning

The problems of heuristic programming (Minsky 1961)

Search – through some large space of solution attempts

Pattern-recognition – restricting to appropriate methods

Learning – by directing search according to experience

Planning – by analysing and further directing search

Induction – for general purpose intelligent machines

Pattern-recognition (Minsky)

- Classify problem situations into categories associated with the domains of effectiveness of the machine's methods
- Extract heuristically significant features of objects
- Define useful properties and make them resistant to noise
- Combine many properties to form a recognition system

Normalization for pattern-recognition

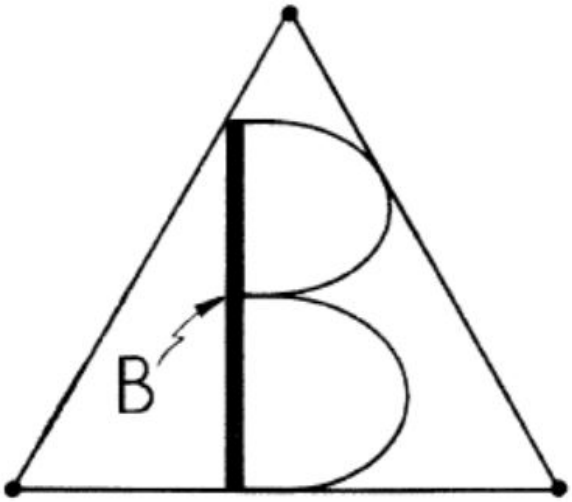


Fig. 2—A simple normalization technique. If an object is expanded uniformly, without rotation, until it touches all three sides of a triangle, the resulting figure will be unique, and pattern-recognition can proceed without concern about relative size and position.

Property functions for pattern-recognition

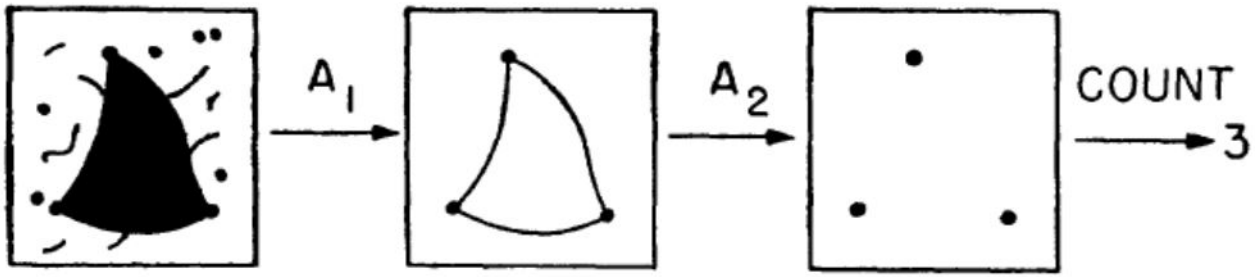


Fig. 5—An arbitrary sequence of picture-transformations, followed by a numerical-valued function, can be used as a *property* function for pictures. A_1 removes all points which are not at the edge of a solid region. A_2 leaves only vertex points—at which an arc suddenly changes direction. The function C simply counts the number of points remaining in the picture. All remarks in the text could be generalized to apply to properties, like A_1A_2C, which can have more than two values.

Learning

- The basic learning heuristic is to use successful methods
- Advanced learning is meta-level learning how to learn (e.g., Friedberg's program-writing program)
- Success-reinforced models are averaging models
- Underdetermined success assessment problem, managed by local reinforcement of partial goals in hierarchies

Learning by reinforcement (Minsky)

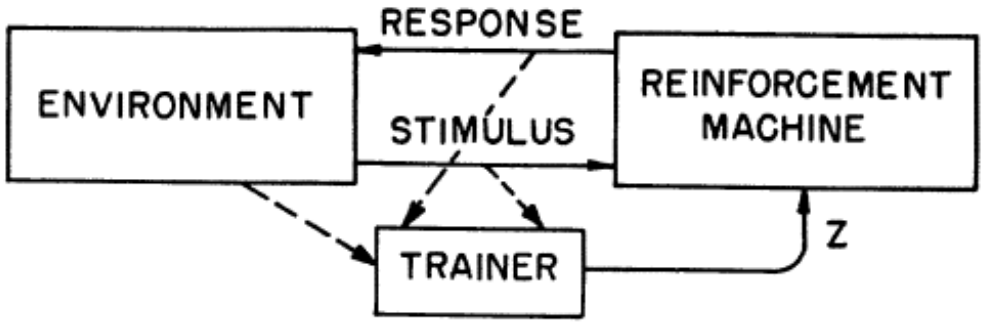
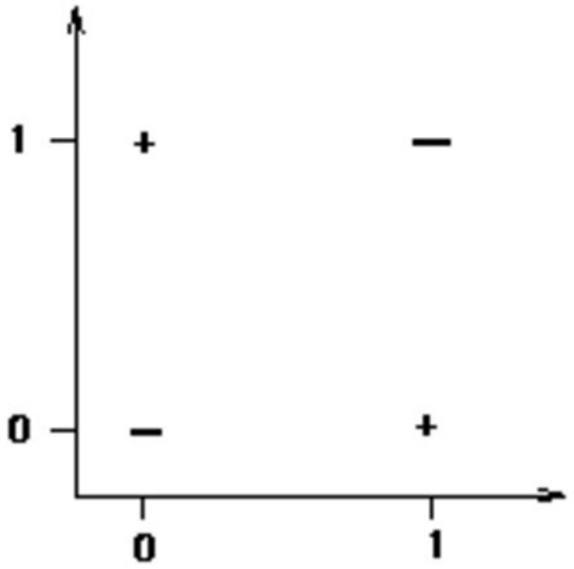


Fig. 8—Parts of an "operant reinforcement" learning system. In response to a stimulus from the environment, the machine makes one of several possible responses. It remembers what decisions were made in choosing this response. Shortly thereafter, the Trainer sends to the machine positive or negative reinforcement (reward) signal; this increases or decreases the tendency to make the same decisions in the future. Note that the Trainer need not know how to solve problems, but only how to detect success or failure, or relative improvement; his function is selective. The Trainer might be connected to observe the actual stimulus-response activity or, in a more interesting kind of system, just some function of the state of the environment.

Single-layer simple machines disappoint

XOR
is an example of a function that cannot be learned by a perceptron:
let + correspond to output 1 and – to output 0, then XOR is not linearly separable



Semantic models and time (Elman)

- "... connectionist representations may have a functional compositionality without being syntactically compositional".
- Time could be represented implicitly, by the effect it has on processing, and not explicitly, as an additional dimension of the output
- Time series, temporal logic, event-based systems, semaphores,...
- How do you capture the dynamics?

Learning units and layers

- Networks can be augmented at the input level by additional units called ‘context units’
- A ‘hidden unit’ interact exclusively with nodes internal to the network, and not the outside world
- Jordan (1986) and Elman (1990) considered hidden layers as parts of recurrent networks

Induction (Elman)

- "... by themselves, the simple learning systems are useful only in recurrent situations; they cannot cope with any significant novelty. Nontrivial performance is obtained only when learning systems are supplemented with classification or pattern-recognition methods of some inductive ability".
- Extract heuristically significant features of objects

Perceptron convergence theorem (Minsky and Papert 1988)

If a set of examples are learnable (100% correct classification), the Perceptron learning rule will find the necessary weights

- in a finite number of steps
- independent of the initial weights

The rule does gradient descent search in weight space, so if a solution exists, gradient descent is guaranteed to find an optimal solution for any 1-layer neural network

C.R. Dyer, Univ of Wisconsin (remix)

The start of learning theory (Vapnik)

In 1962 Novikoff proved the first theorem about the perceptron (Novikoff, 1962). This theorem actually started learning theory. It asserts that if

(i) the norm of the training vectors z is bounded by some constant R ($|z| \leq R$);

(ii) the training data can be separated with margin ρ:

$$\sup_{w} \min_{i} y_i (z_i \cdot w) > \rho;$$

(iii) the training sequence is presented to the perceptron a sufficient number of times,

then after at most

$$N \leq \left[\frac{R^2}{\rho^2} \right]$$

corrections the hyperplane that separates the training data will be constructed.

Beyond perceptrons

Perceptrons can only learn linearly-separable functions

Multi-layer feedforward networks generalize 1-layer networks to n-layer networks (Rumelhart, Hinton, and Williams 1986)

- Partition units into n+1 layers, such that layer 0 contains the input units, layers 1, ..., n-1 are the hidden layers, and layer n contains the output units
- Each unit in layer k, k=0, ..., n-1, is connected to all of the units in layer k+1
- Connectivity means bottom-up connections only, with no cycles, hence 'feedforward'
- Programmer defines the number of hidden layers and the number of units in each layer (input, hidden, and output)

C.R. Dyer, Univ of Wisconsin (remix)

Programming multi-layer machines

- Programmer specifies numbers of units in each layer and connectivity between units, so the only unknown is the set of weights associated with the connections
- Supervised learning of weights from a set of training examples, given as I/O pairs
- Each pass through all of the training examples is one 'epoch'

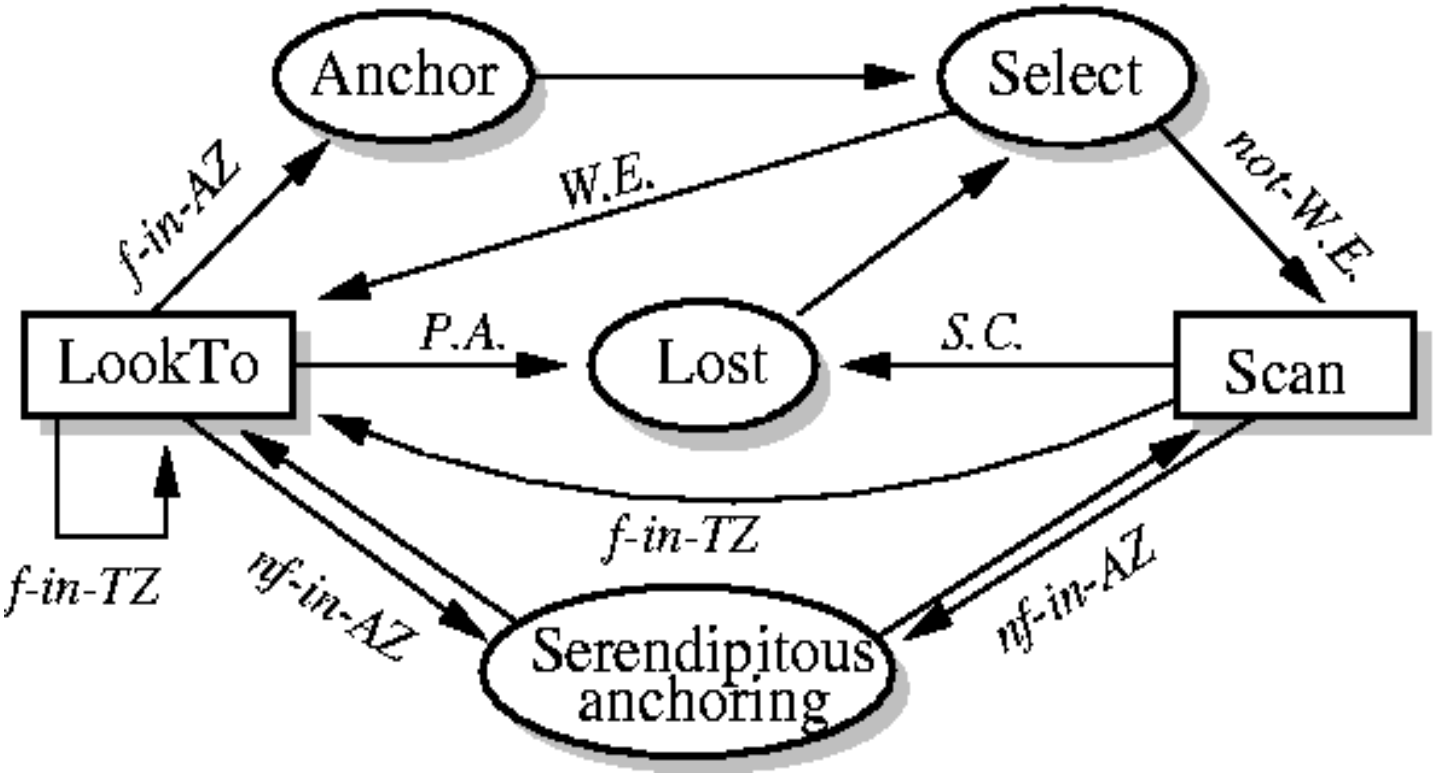
Algorithm:

1. Initialize the weights in the network (usually with random values)
2. repeat until stopping criterion is met
   foreach example e in training set do
   a) O = neural-net-output(network, e)
   b) T = desired (Teacher) output
   c) update-weights(e, O, T)

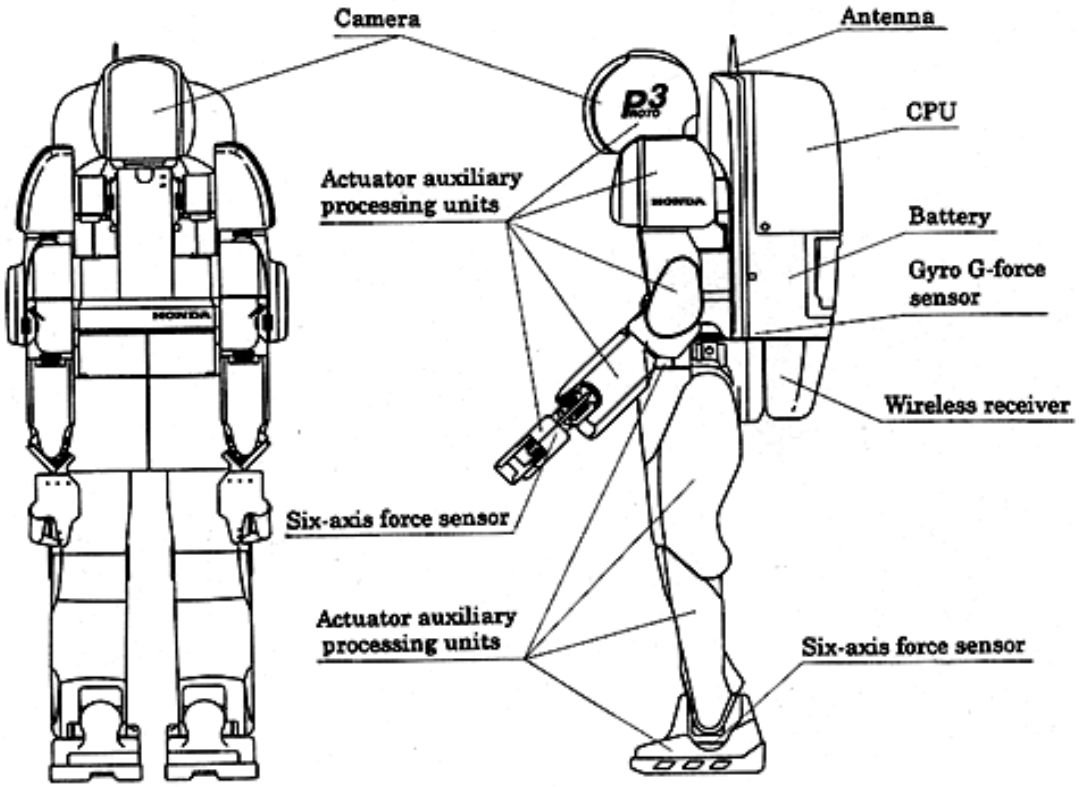
C.R. Dyer, Univ of Wisconsin (remix)

Loebner Prize at Bletchley Park

TeamSweden tracking and anchoring



The learning machine as a humanoid



Real-time decisions

Questions?

Or play DOTA 2 1v1

https://www.youtube.com/watch?v=l92J1UvHf6M

Or play DOTA 2 5v5