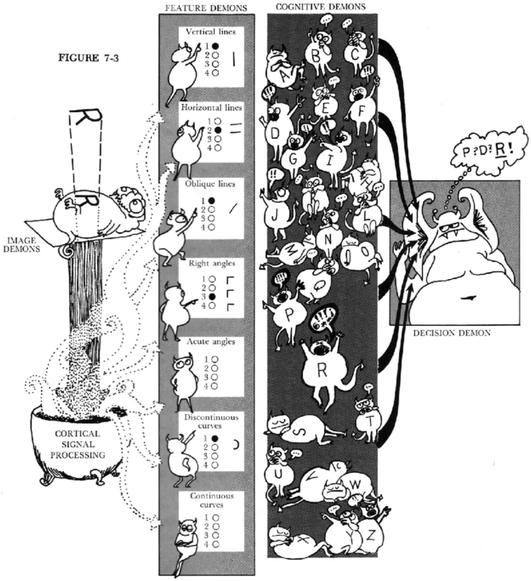
Backpropagation

Multi-layer perceptrons (MLPs)

How to solve the XOR problem ...

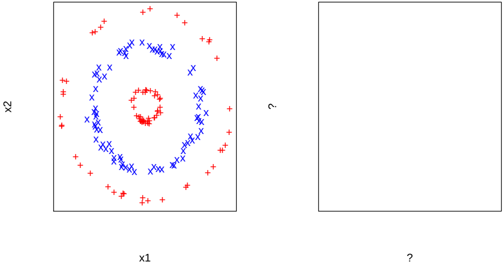
The importance of features / 1



(Lindsay and Norman's view of Selfridge's Pandemonium model, 1959).

The importance of features / 2

- Find the right features to make the task solvable:



- Engineering features by hand is hard.
- Neural networks learn features that they find important.

How many layers of features do you need?

One hidden layer is all you need **in theory** to make a "universal approximator" (Cybenko 1989; Hornik 1991).
Online example:
`http://neuralnetworksanddeeplearning.com/chap4.html` shows how to approximate 1-d and 2-d function with one layer of (many) hidden units.
With linear transfer functions how many layers do we need?

Neural networks and linear algebra

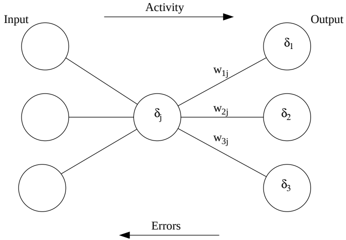
- Activation of a layer of neurons stored in a **vector**.
- Synapses from one layer to another stored in a **weight matrix**: $\mathbf{W}_{ji}$ is strength of connection from unit i in one layer to unit j in the next layer.
- "The single key fact about vectors and matrices is that each vector represents a point located in space, and a matrix moves that point to a different location. Everything else is just details." (Stone 2019, Appx. C).

Learning in multi-layer perceptrons

How to solve the credit assignment problem? i.e. what is the "delta" for hidden units, that have no desired output?

$$\delta_j = g'(h_j) \sum_i w_{ij} \delta_i$$

h_j is total input to unit j; g' is 1st derivative of activation function.



No guarantee (unlike PCT) that this will converge due to local minima.

Application: solving XOR...

DEMO in live class.

How to assess for over fitting vs under fitting

- **Underfitting**: can network solve problem?
- **Overfitting**: network too focused on learning (perhaps by rote) the training examples.
- **Generalisation**: how well does network perform on inputs not seen during learning?
- Where is the “Goldilocks spot”?
- Run on **validation set** during learning. Plot error as a function of training time (epochs). Assess after on test set.
- Other approaches (k-fold validation) also feasible. Be careful about time-dependencies!

Some terms

1. **Online learning**: learn after every input. (each **iteration**). Sometimes called **stochastic gradient descent** as approximating gradient with one sample at a time.
2. **Batch learning**: wait until all training samples have been presented (each **epoch**).
3. **Mini batch**: break data into several groups (make sure each is balanced).

Applications: family trees (Hinton 1986; Paccanaro and Hinton, 2000).

- How to predict family trees? (person X) (relationship) (who?)
- e.g. (Charlotte) (has aunt) (who) $\Rightarrow$ Jennifer or Margaret
- 2 family trees of 12 people = 24 people.
- 12 possible relationships:
  husband, wife, son, daughter, father, mother
  brother, sister, nephew, neice, uncle, aunt
- 104 relationships; 100 used in training; 4 for testing.

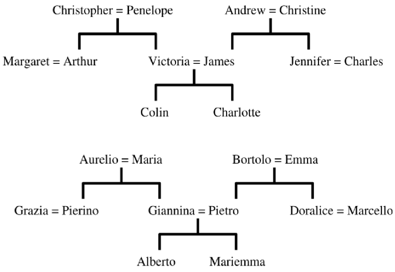
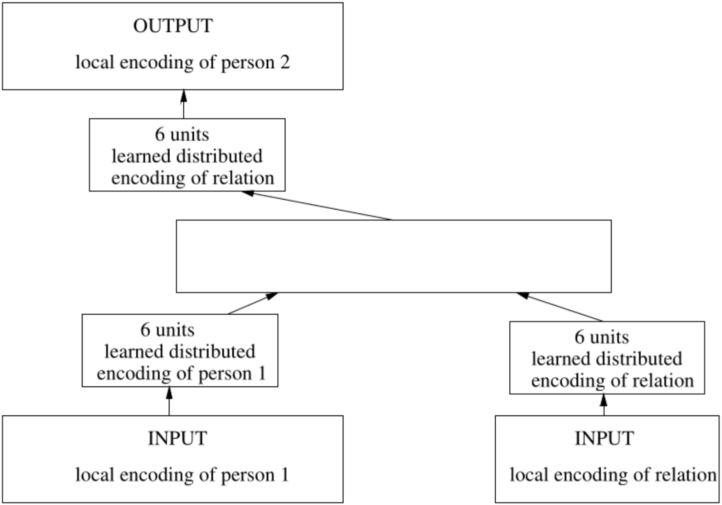


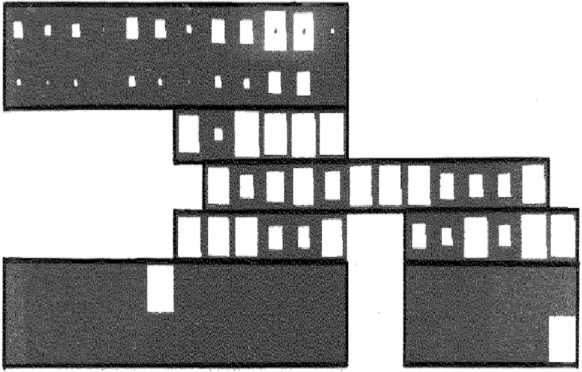
Figure 2: Two isomorphic family trees. The symbol "=" means "married to"

Family tree architecture

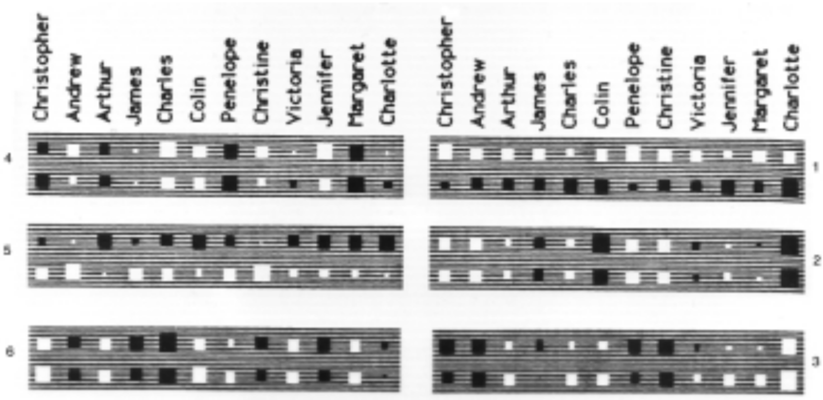


After training

When tested on: (Colin) (has Aunt). See next slide for interpretation of outputs.



What are the hidden units doing?



Generalisation

Very small data set, but normally got at least 2 (of 4) test set examples correct.

Applications: NET TALK

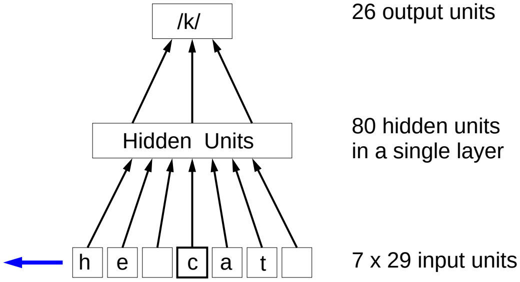
NetTalk: Sejnowski and Rosenberg (1987)

How do humans learn to pronounce words? Develop mapping from letters to phonemes/stress.
Corpus chosen from either (a) 1000 most common words from Pocket Dictionary (b) young child's informal speech.
Context of letter important: mapping from letter to pronounciation dependent on context:

phoneme	word	articulation features
/A/	b*i*te	medium, tensed, front2+central1
/I/	b*i*t	high, front1

Too many words (even with approx 18,000 weights) to just produce lookup table – must extract principles.

NetTalk: architecture

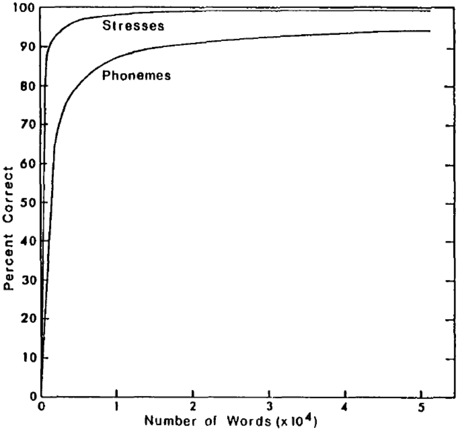


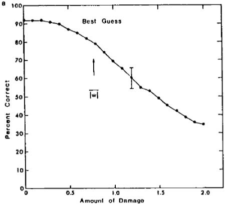
Input: each letter encoded using 1 of 29 units (26 + 3 for punctuation). Central letter is processed, using surrounding 3 letters each side to provide context.

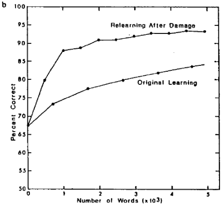
Output: distributed representation across 21 features including vowel height, position in mouth; 5 features for stress.

NetTalk: performance

- Stage-like progression of behaviour:
    1. distinction between vowels and consonants: but same vowel for all vowels, and same consonant for all consonants. (babbling)
    2. recognition of word boundaries.
    3. pseudo words
    4. good performance (90%)
- Power-law like learning (similar to humans)
- Robust to weight damage; rapid recovery.
- Generalisation: $\approx 80\%$







NetTalk: demo

Demonstration of NetTalk:
`http://cnl.salk.edu/Media/nettalk.mp3`

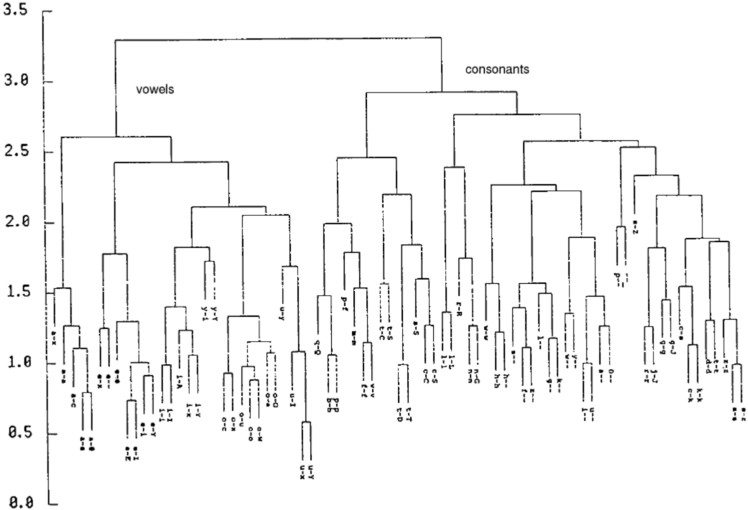
- Part 1 : learning from zero weights (0:37).
- Part 2 : learning after 10,000 iterations (3:18).
- Part 3 : performance on unseen text during learning (5:02).

NB: Similar architecture of moving input window used to predict protein structure (α helix, β*sheet*, other) of central part of 13 amino acids (Qian & Sejnowski, 1988).

NetTalk: Hidden unit analysis

What features are the network extracting? Compute 80-d vector of average activity for given input–output pair (of which there are 79).

We will return to this, in dimensionality reduction.



Summary

1. We have now seen the work of two of the leading pioneers in the field: Terry Sejnowski and Geoff Hinton.
2. Also known as “the Lennon and McCartney of neural networks” (Jim Stone).