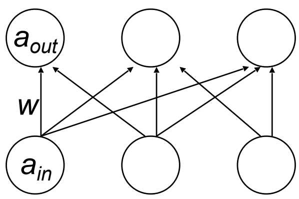
Connectionist Language Processing

Lecture 3: **Multi-layer Networks**

Matthew W. Crocker
crocker@coli.uni-sb.de
Harm Brouwer
brouwer@coli.uni-sb.de

"Perceptrons" [Rosenblatt 1958]

- Perceptron: a simple, one-layer, feed-forward network:



$$\text{netinput}_{out} = \sum_{in} w \cdot a_{in}$$

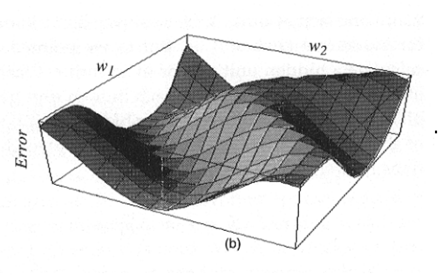
- Binary threshold activation function:

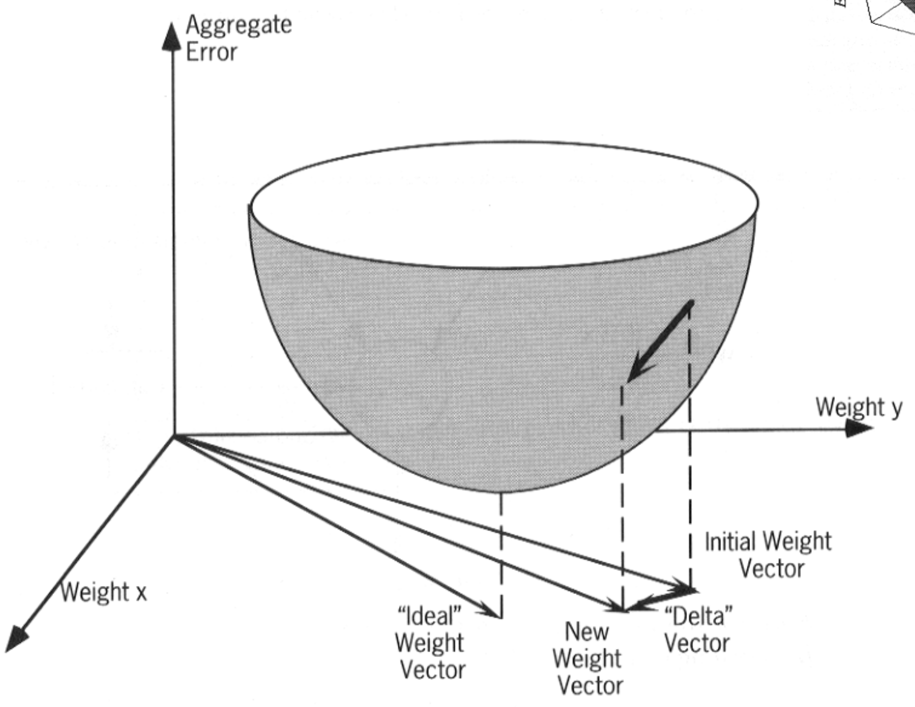
$$\begin{aligned} a_{out} &= 1 \text{ if netinput}_{out} > \theta \\ &= 0 \text{ otherwise} \end{aligned}$$

- Learning: the perceptron convergence rule
  - Two parameters can be adjusted:
    - The threshold
    - The weights

$$\text{The error, } \delta = (t_{out} - a_{out})$$
$$\Delta\theta = -\varepsilon\delta$$
$$\Delta w = \varepsilon\delta a_{in}$$

Visualising the error





Gradient descent continued

- We need calculus to allow us to determine how the error varies when a particular weight is varied:

$$\Delta w = -\varepsilon \frac{\partial E}{\partial w}$$

Slope: Rate of change of *E*, with *w*

$$\Delta w = -\varepsilon \frac{\partial (t_{out} - a_{out})^2}{\partial w}$$

Error = $(t_{out} - a_{out})^2$

$$\Delta w = -\varepsilon \frac{\partial [t_{out} - F(\sum_{in} w \cdot a_{in})]^2}{\partial w}$$

Derivative of the activation function with respect to w, i.e. its slope

$$\Delta w = 2\varepsilon [t_{out} - F(\sum_{in} w \cdot a_{in})] \cdot F'(\sum_{in} w \cdot a_{in}) \cdot a_{in}$$

$$\Delta w = 2\varepsilon \delta F^{*} a_{in}$$

$$\begin{bmatrix} \delta = (t_{out} - a_{out}) \\ F^{*} = \text{slope of the activation function} \end{bmatrix}$$

For the logistic: $F^{*} = a_{out}(1 - a_{out})$

$$F^{*} = a_{out}(1 - a_{out})$$

Summary – Learning Rules

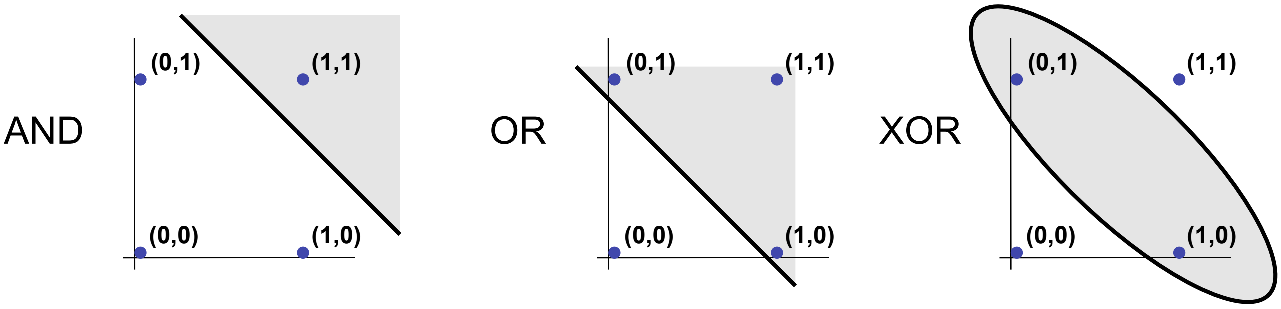
- Perceptron convergence rule
- Delta rule
  - Depends on the (slope of the) activation function
- For 2-layer networks using these rules:
  - A solution will be found, if it exists
- How do we know if network has learned successfully?

Summary – Error

- For learning, we use (t_{out} - a_{out}) for each output unit, to change weights
- To characterise the performance of the network as a whole, we need a measure of global error:
  - Across all output units
  - Across all training patterns
- One possible measure is RMS
  - Another is entropy: doesn't matter too much, since we only need to know if performance is improving or deteriorating on a relative basis
  - But, low overall error doesn't always mean the network has learned successfully!

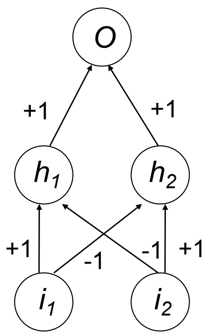
Linear Separability

- Single layer networks, including perceptrons, can only learn input-output mappings that are “linearly separable”.



Solving XOR with hidden units

- Consider the following network:
    - two-layer, feedforward
    - 2 units in a “hidden” layer
    - Hidden and output units are threshold units: $\theta = 1$

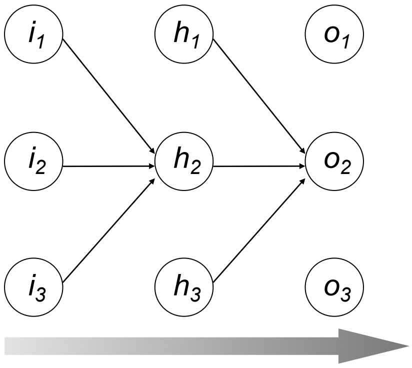


Input	Hidden		Target
	h_1	h_2	
0 0	0	0	0
1 0	1	0	1
0 1	0	1	1
1 1	0	0	0

- Representations at hidden layer:
- Problem: current learning rules cannot be used for hidden units:
    - Why? We don’t know what the “error” is at these nodes (no target)
    - “Delta” requires that we know the “target” activation

$$\Delta w = 2\varepsilon \delta F^* a_{in}$$

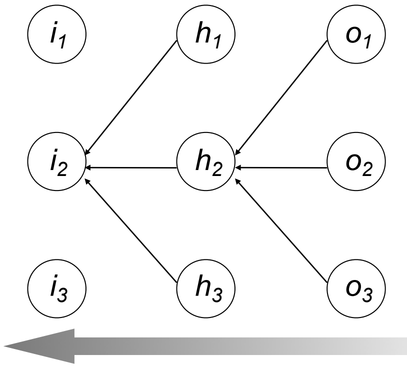
Backpropagation of Error



(a) Forward propagation of activity:

$$\text{net}_{out} = \sum w_{oh} \cdot a_{hidden}$$

$$a_{out} = f(\text{net}_{out})$$



(b) Backward propagation of error:

$$\text{err}_{hidden} = \sum w_{oh} \cdot \delta_{out}$$

$$\delta_{hidden} = f'(\text{net}_{hidden}) \cdot \text{err}_{hidden}$$

Generalized Delta Rule

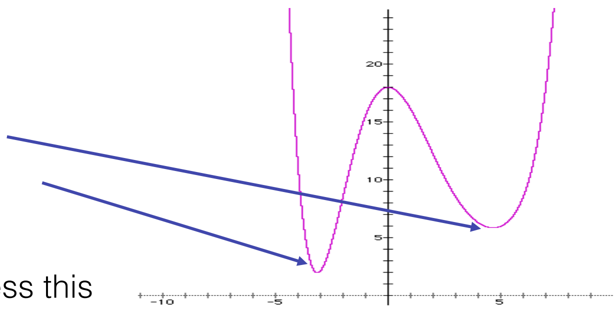
$$\Delta w_{ij} = \varepsilon \delta_i a_j$$

For output nodes: $\delta_k = \sigma'(net_k)(t_k - a_k)$

For hidden nodes: $\delta_i = \sigma'(net_i) \sum_k w_{ki} \delta_k$

where, $\sigma'(net_i) = a_i(1 - a_i)$

- Multi-layer networks can, in principle, learn any mapping function:
  - Not just linearly separable ones
- But while there exists a solution for any mapping problem
  - backpropagation is not guaranteed to find it
- Why? Local minima:
  - Backprop can get trapped here
  - Global minimum (solution) is here
  - There are various means to address this



Example of Backpropagation

- Consider the following network, containing a single hidden node
- Calculate the weight changes for both layers of the network, assuming learning rate ε = 0.1 and targets of: 1 1

The generalised Delta rule:

$$\Delta w_{ij} = \varepsilon \delta_i a_j$$

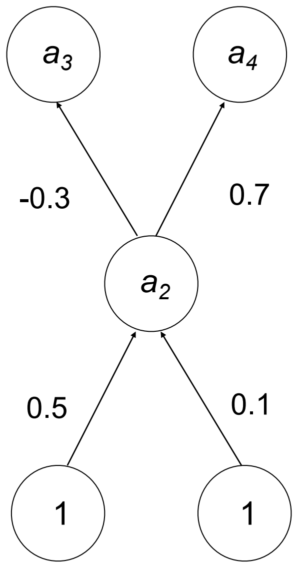
For output nodes:

$$\delta_k = \sigma'(net_k)(t_k - a_k)$$

For hidden nodes:

$$\delta_i = \sigma'(net_i)\sum_k \delta_k w_{ki}$$

where, $\sigma'(net_i) = a_i(1 - a_i)$



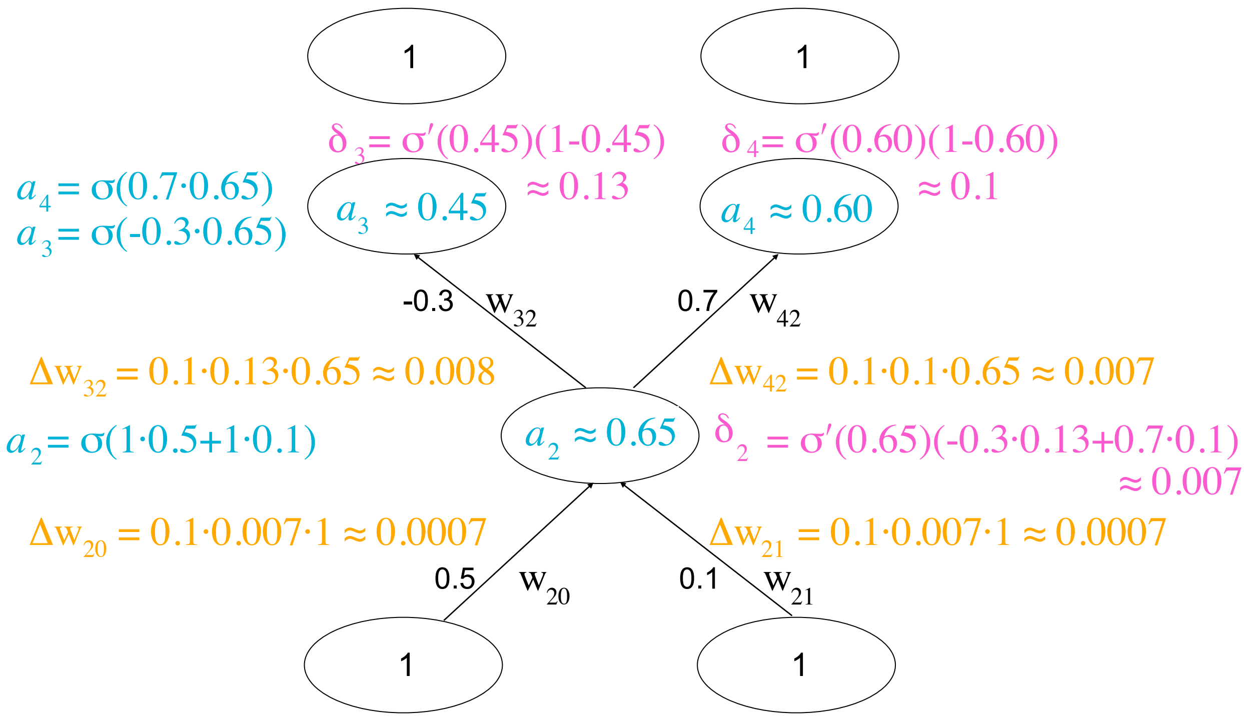
Forward and Backpropagation

$$\Delta w_{ij} = \varepsilon \delta_i a_j$$

For output nodes: $\delta_k = \sigma'(net_k)(t_k - a_k)$

For hidden nodes: $\delta_i = \sigma'(net_i)\sum_k w_{ki}\delta_k$

where, $\sigma'(net_i) = a_i(1 - a_i)$



Learning lexical mappings

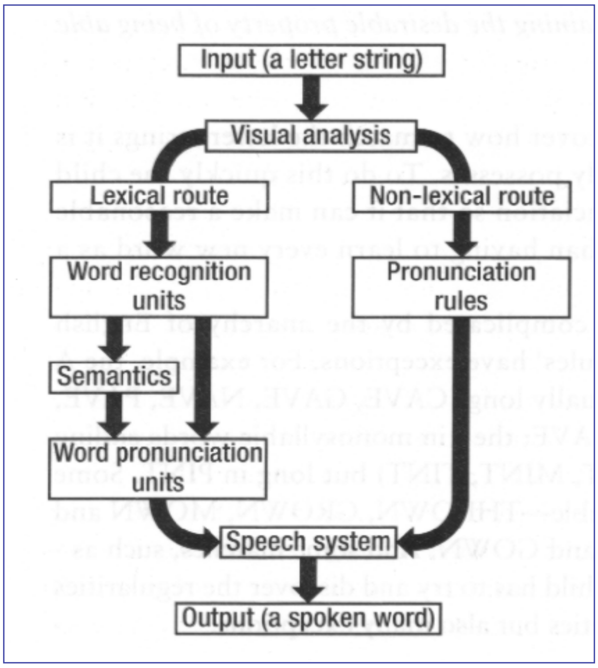
- **Reading aloud:** Mapping Orthography to Phonology
- **English past-tense:** Forming the past tense from the present
- *Dual route* accounts of exceptional vs regular forms
  - Evidence: double dissociation in acquired dyslexics
- Connectionist account: a single mechanism
  - Good performance on known and unknown words
  - Models (normal) human behaviour
  - Importance of input and output representations
  - Double dissociations?

Reading Aloud

- **Task**: produce correct pronunciation for a word, given its printed form
- Suited to connectionist modeling:
  - Need to learn mappings from one domain (orthography) to another (sound)
  - Multi-layer networks are good at this, even when mappings are arbitrary
  - Human learning is similar to network learning:
    - I.e. learning takes place gradually over time
    - Incorrect attempts are often corrected
- If a network can't model this linguistic task successfully, it would be a serious blow to connectionist modeling. But …

Dual Route Model

- The standard model of reading posits two independent routes leading to pronunciation of a word, because …
  - People can easily pronounce words they have never seen:
    - SLINT or MAVE
  - People can pronounce words which break the “rules”:
    - PINT or HAVE
- One mechanism uses general rules for pronunciation
- The other mechanism stores pronunciation information with specific words



Behaviour of Dual-Route Models

- Consider: MINT, PINT, and KINT
- MINT is a word:
  - Can be pronounced using the “rule-based” mechanism
  - But also exists in the lexicon, so can be pronounced by the “lexical” route
- PINT is a word, but irregular
  - Can only be correctly pronounced by the lexical route
  - Otherwise, it would rhyme with MINT
- KINT is not a word:
  - No entry in the lexicon
  - Can only be pronounced using the “rule-based” mechanism
  - So should rhyme with MINT

Evidence for Dual-Route Model

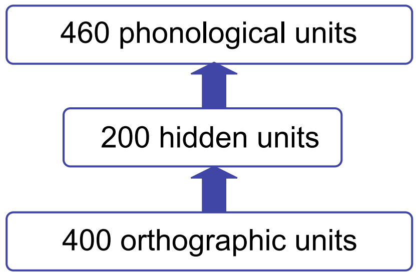
- Evidence from neuropsychology shows different patterns of behaviour for two types of brain damage that are acquired after learning
- Phonological dyslexia
  - **Symptom**: Read words without difficulty, but cannot produce pronunciations for non-words
  - **Explanation**: Damage to rule-based route; lexical route intact
- Surface dyslexia:
  - **Symptom**: Can pronounce words and non-words correctly, but tend to regularise irregulars
  - **Explanation**: Damage to the lexical route; rule-based route intact
- All Dual-Route models share:
  - A lexicon for known words, with specific pronunciation information
  - A rule mechanism for the pronunciation of unknown words

Towards a Connectionist Model

- It is unclear how a connectionist model could naturally implement a dual-route model:
  - No obvious way to implement a lexicon to store information about particular words; storage is typically distributed
  - No clear way to distinguish “specific information” from “general rules”; only one uniform way to store information: connection weights
- Seidenberg & McClelland (1989): a standard 2-layer feedforward model
  - Trained to pronounce all the monosyllabic words of English
  - Learning is implemented using the backpropagation algorithm

Seidenberg and McClelland (1989)

- 2-layer feed-forward model:
  - Distributed representations at input and output
  - Distributed knowledge within the net
  - Gradient descent learning
- Input and Output
  - Inputs are activated by the letters of the words
    - 20% activated, on average
  - Outputs represent the phonological features
    - 12% activated, on average
  - Encoding of features does not affect the success
- Processing: Node activation is determined using the logistic function



$$\text{netinput}_i = \sum_j a_j w_{ij} + \text{bias}_i$$

Training the Model

- Learning
  - Weights and bias are initially random
  - Words are presented and outputs are computed
  - Connection weights are adjusted based on backpropagation of error
- Training: All monosyllabic words of 3 or more letters (about 3000) words
  - In each epoch, a subset was presented: frequent words appeared more often
    - Over 250 epochs, (THE) was presented 230 times, least common 7 times
- Performance
  - Outputs were considered correct if closer to the correct pronunciation than that of any other word
  - After 250 epochs, accuracy was 97%

Results: Seidenberg & McClelland

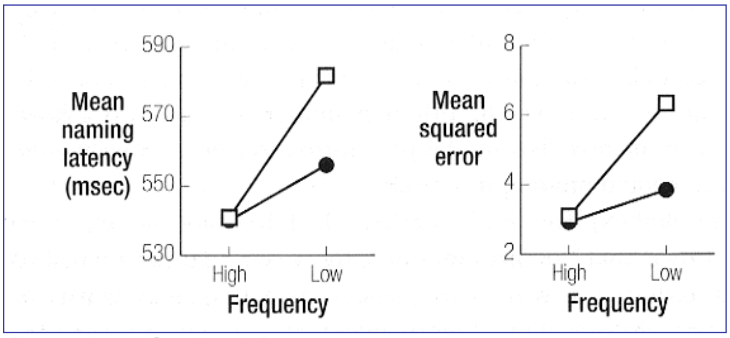
- The model does successfully learn to map most regular and irregular word forms to their correct pronunciation
  - It does this without separate routes for lexical or rule based processing
  - There is no word specific memory
  - It does not perform as well as humans in pronouncing non-words
- **Naming Latency:** Adult reaction times for naming a word is a function of variables such as word frequency and spelling regularity
  - The current model cannot directly mimic latencies
- If we **relate the output error score to latency**, where phonological error score is the difference between the actual pattern and the correct pattern
  - Hypothesis: high error should correlate with longer latencies

Word Frequency Effects

- Common words are pronounced more quickly than uncommon words
  - This is true for almost all aspects of human information processing
- Conventional (localist) explanation:
  - Frequent words require a lower threshold of activity for “the word recognition device” to “fire”
  - Infrequent words require a higher threshold of activity
- In the Seidenberg & McClelland model, naming latency is modeled by error:
  - Word frequency is reflected in the training procedure
  - Phonological error is reduced by training, thus lower for high frequency words
- The explanation of latencies in terms of error follows directly from the network’s architecture and the training regime

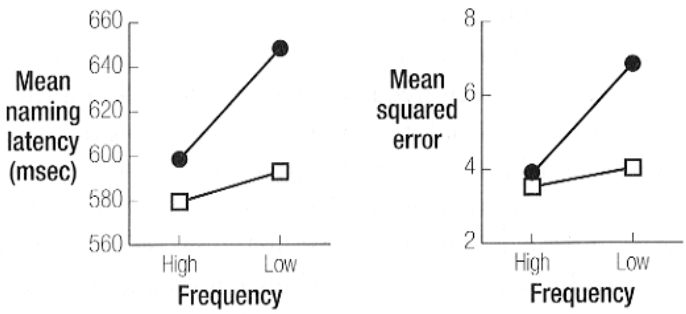
Frequency x Regularity

- In addition to faster naming of frequent words, human subjects exhibit:
  - Faster pronunciation of regulars (e.g GAVE) than irregulars (e.g. HAVE)
  - But this interacts with frequency: it is only observed with low frequency words
- For regulars (filled circle) we observe a small effect of frequency
  - It takes slightly longer to pronounce the low frequency regulars
- For irregulars (open square) we observe a large effect of frequency
- The model precisely mimics this pattern:
- 2-route: Lexical route wins faster for high frequency words, while confusion of the lexical and rule outcome requires resolution for the irregular words



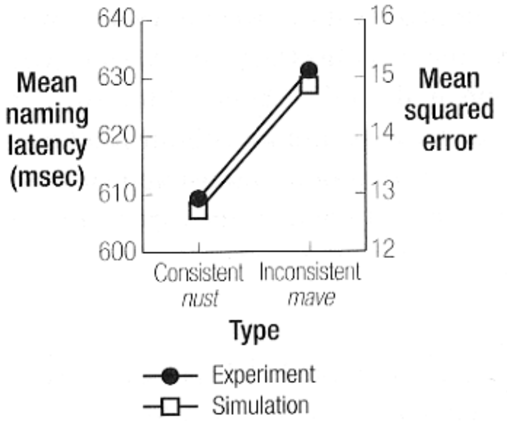
Frequency x Neighborhood Size

- The **neighborhood size** of a word is the number of words that differ by changing one letter
- Neighborhood size has also been shown to affect naming latency, as with regularity:
  - Not much influence for high frequency words
  - Low frequency words with small neighborhoods (filled circles) are read much more slowly than words with large neighborhoods (open squares)
- Shows "cooperation" of the information learnt in response to different (but similar) inputs
- Again, the connectionist model directly predicts this
- The 2 route model requires a more ad hoc explanation, grouping across localist representations of the lexicon



Spelling-to-Sound Consistency

- Consistent spelling patterns: _UST
  - All words have the same pronunciation
- Inconsistent patterns are those with more than one: _AVE
- Observation: adult readers produce pronunciations more quickly for non-words derived from consistent patterns (NUST) than from inconsistent patterns (MAVE)
- This is difficult for 2-route models:
  - Since both are processed by the non-lexical route
  - Consistent and inconsistent rules would need to be distinguished
- The error in the connectionist model predicts this latency effect perfectly



Seidenberg & McClelland (1989)

- The model is a single mechanism with no lexical entries or explicit rules
- Response to an input is a function of the network's entire experience
  - Reflects previous experience on a particular word
  - Experience with words resembling that string
- E.g. specific experience with HAVE is sufficient to overcome the general information that _AVE is usually a long vowel
- The network can produce a plausible pronunciation for MAVE, but error is introduced by experience with inconsistent words like HAVE
- Performance: 97% accuracy on pronouncing learned words
  - Models: frequency & interaction with regularity, neighborhood, consistency
- Limitations: It is not as good as humans at
  - Reading non-words (model gets 60%, humans 90%)
  - Lexical decision (FRAME is a word, but FRANE is not)