Computational Psycholinguistics

Lecture 12: Connectionist Models of Language Processing

Afra Alishahi

February 2, 2009

(based on slides by Matthew Crocker and Marshall Mayberry)

Connectionist Modeling

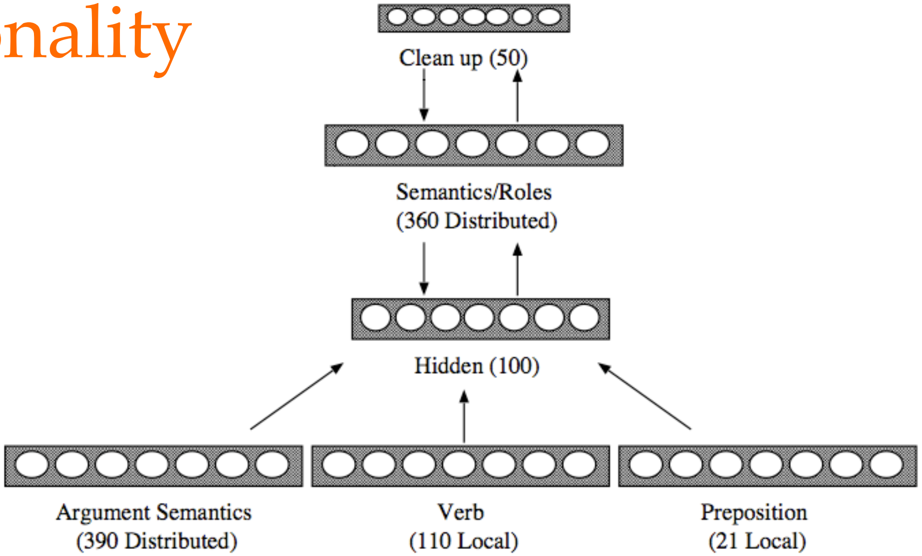
- Connectionism was proposed as an alternative to the symbolic accounts of information processing
  - Motivation: design computers inspired by brain
  - Key ideas: distributed, implicit representations; dense connectivity; communication of 'real values' not 'symbols'; single mechanism for rules and exceptions
- A functionalist assumption of language:
  - knowledge of language develops in the course of learning how to perform primary communicative tasks of comprehension and production

Overview

- An input stimulus causes a pattern of activation on the first layer
  - Activations are then propagated through the network
  - Weights determine the influence of unit on each other
  - The output is the pattern of activation on final layer
- Learning aims to reduce the discrepancy between actual and desired output patterns of activation
  - Delta rule changes the weights of successive epochs
  - Training is complete when error is sufficiently reduced

Representing Time

- Many cognitive functions involve processing sequences of inputs/outputs over time:
  - Sequences of sounds to produce a particular word
  - Sequences of words encountered incrementally
- Represent the serial order of the input with the dimensionality of the input vector
- E.g., Allen (1997)



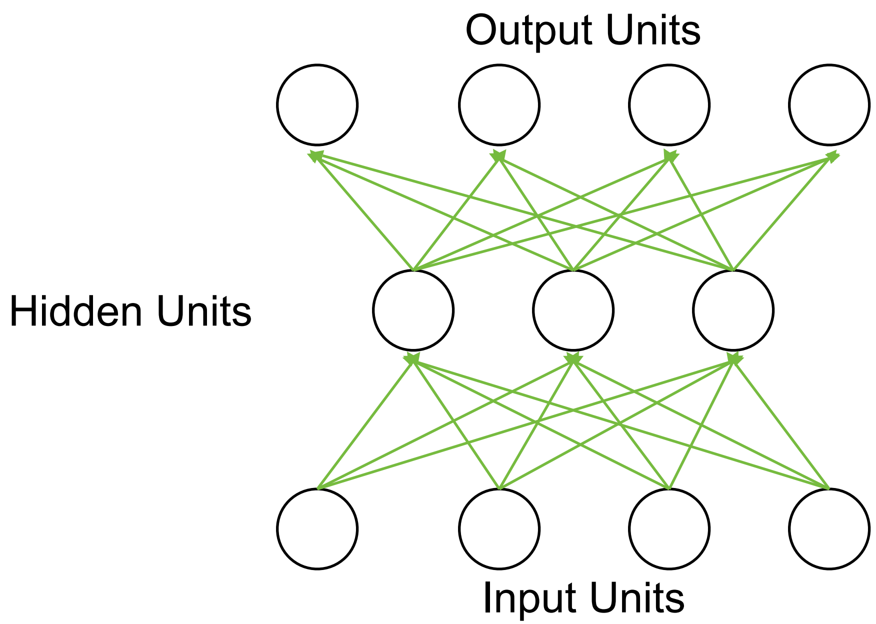
Time as Spatial Order

- Buffering of events before processing, and processing the input all at once
  - Maximum sequence length (duration) is fixed
  - Does not easily distinguish relative versus absolute temporal position, e.g.

    0 1 1 1 0 0 0 0 0
    0 0 0 1 1 1 0 0 0

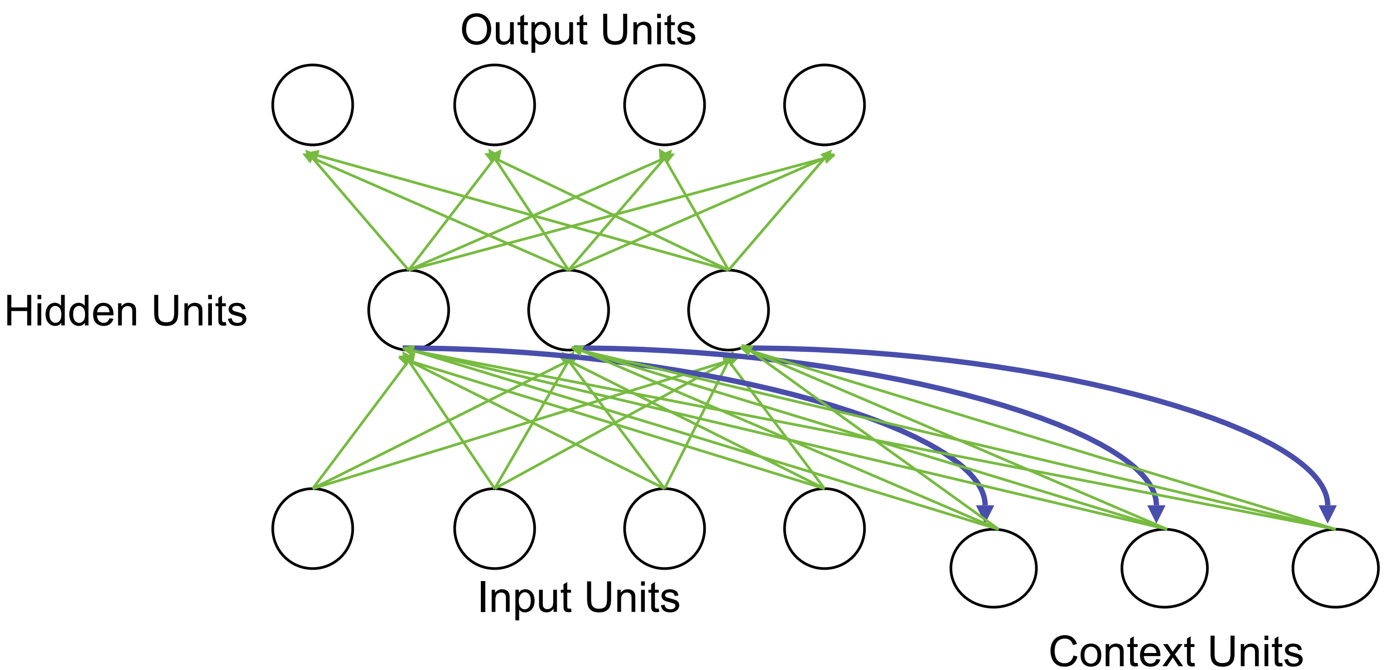
    - Similar patterns are spatially distant
    - Most importantly, in contrast with incrementality
- We need a more general representation of time

Providing Context



- Output depends on the activation pattern in hidden units, which in turn depends on the current input
- Ideally, we want the previous activities to also affect the output

Providing Context



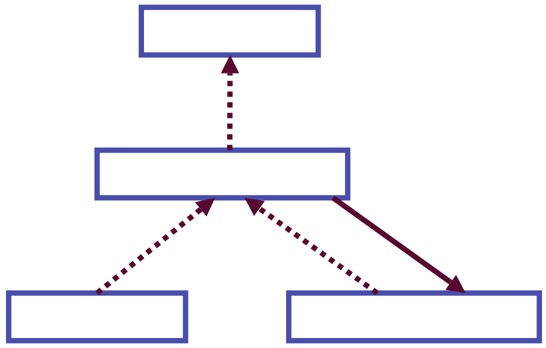
- Simple Recurrent Networks: Elman (1989)
  - keep a copy of the hidden units from the previous step

Simple Recurrent Networks

- Simple Recurrent Networks (SRNs) are trained to predict the next item
    - SRNs can learn any input sequence
- Hidden units are connected to context units:
    - These correspond to states: they remember the state of the network on the previous time step
    - Dynamic memory: identical inputs can be treated differently depending on context

SRNs

- Context units are direct copies of hidden units
  - Connections are one-to-one
  - Weights are fixed at 1.0, and not modified during training



- Connections from context units to hidden units are modifiable
  - Weights are learned just like all other connections
  - Network is trained via the back-propagation learning

Structure in Letter Sequences

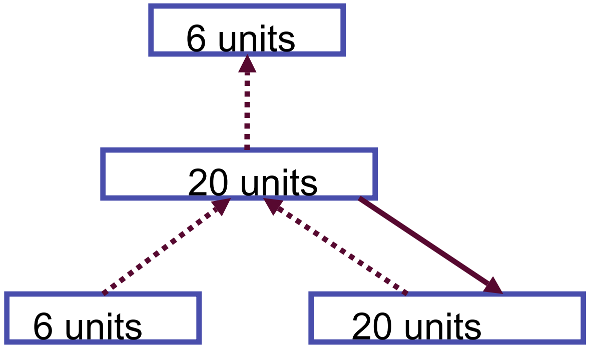
- Training an SRN to learn simple transitions between adjacent letters in a sequence
  - Rules for word formation:
    b → ba   d → dii   g → guuu
  - The 3 consonants were randomly combined to generate a 1000 letter sequence

    dbgbdd… → diibaguuubadiidii…
  - Each letter was converted to a 6 bit representation

	Consonant	Vowel	Interrupted	High	Back	Voiced
b	1	0	1	0	0	1
d	1	0	1	1	0	1
a	0	1	1	0	1	1

Training & Performance

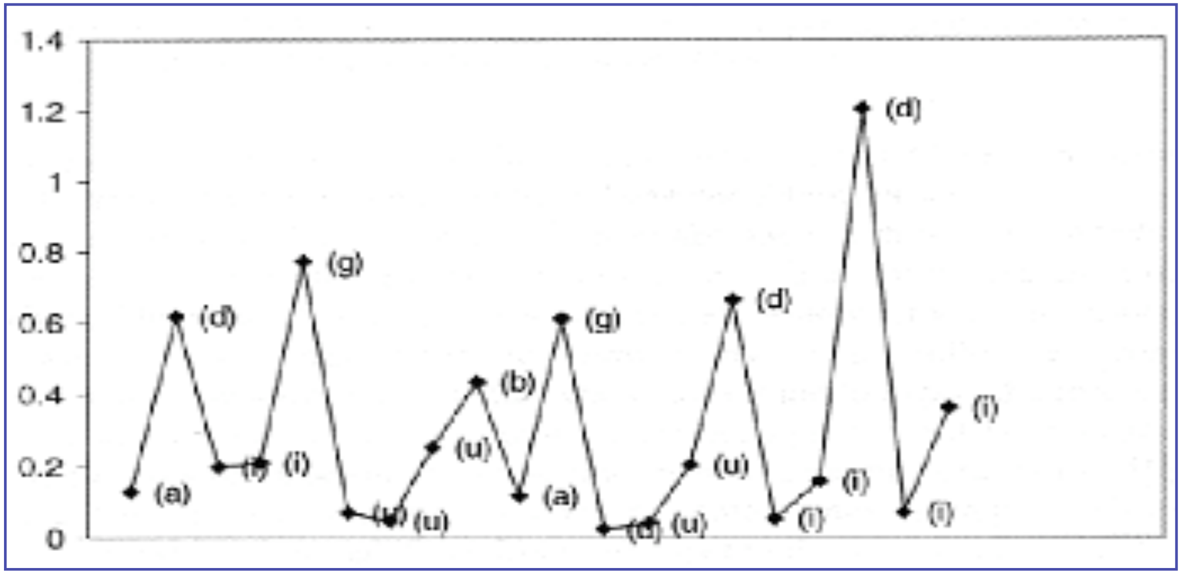
- Architecture:



- Training:
    - Each input vector is presented
    - Network is trained to predict the next input
    - 200 passes through the sequence
- Tested on another random sequence

Error Pattern

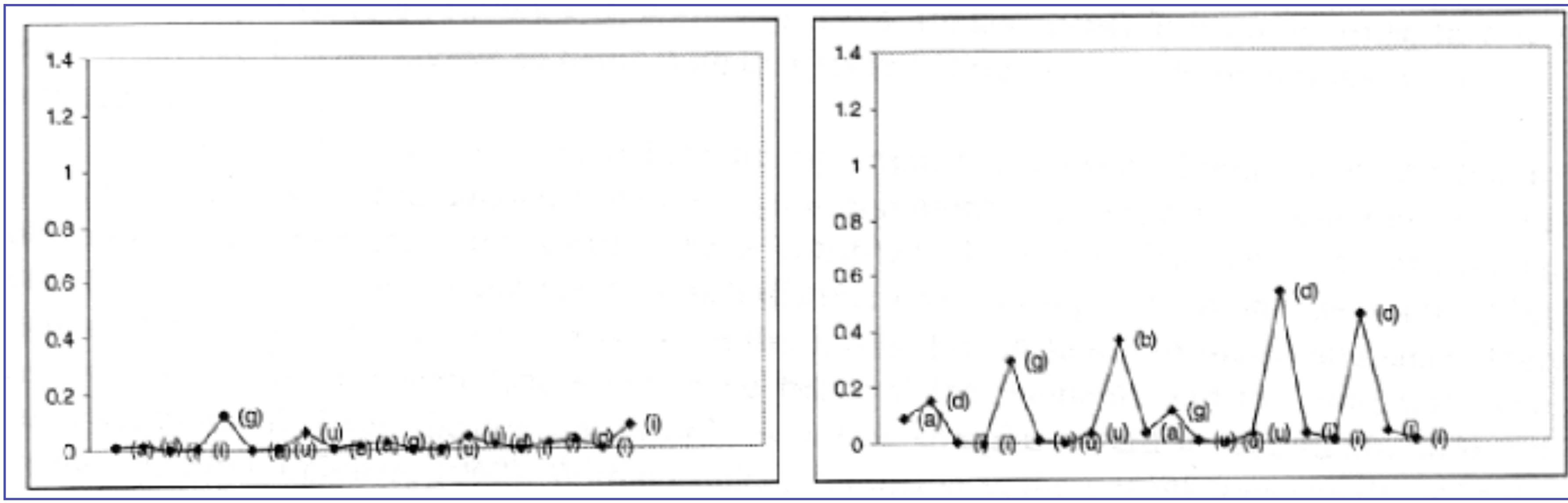
- Global error of the network for the test sequence:



- Predict which (and how many) vowels follow a consonant is easier than predicting consonants
  - Low error on predicting vowels
  - High error on predicting consonants

Deeper analysis of performance

- We can examine the error for the individual bits, e.g. bit 1 (Consonant) and 4 (High)



- All consonants have the same value for feature 1 but not 4
- Network has learned that after the correct number of vowels, *some* consonant is expected

Discovering Word Boundaries

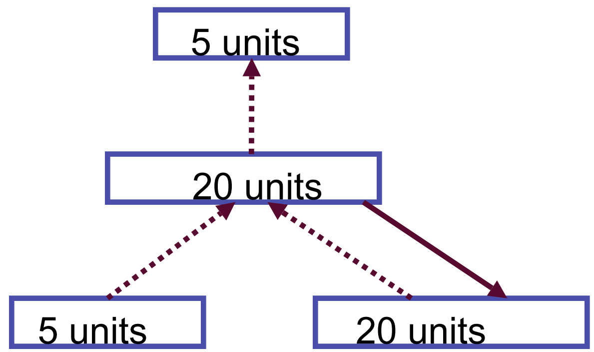
- Existence of words is often taken for granted, but infants face an unsegmented acoustic stream
  - How do they learn to identify word boundaries?
- Simulation: predicting the next sound

  Manyyearsagoaboyandgirllivedbytheseatheyplayedhappily

  - Task: receive a phoneme and predict the next one
- At time *t*, the network knows the current input (phoneme at time *t*) and the results of processing at time *t -1* (context units)

Structure of Network & Input

- Architecture:

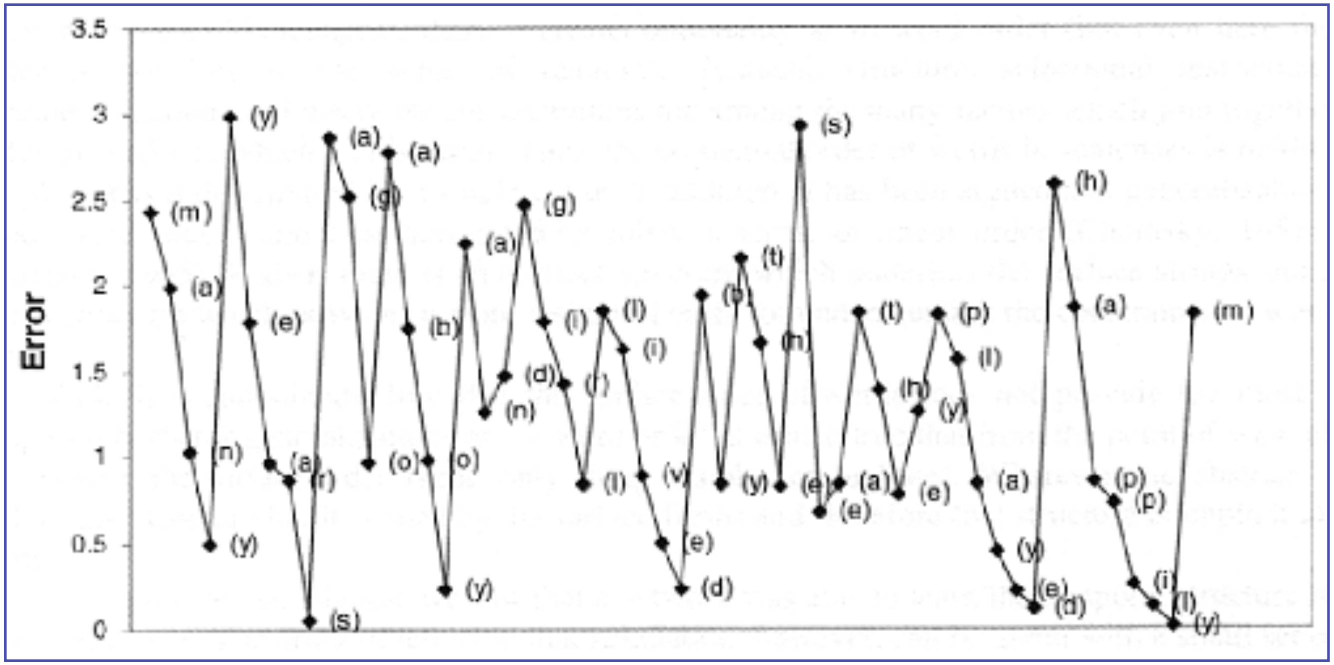


- Input: an approximation of the acoustic signal
  - 200 sentences of length 4 to 9 words, from a lexicon of 15 words
  - Each letter converted to a random 5 bit vector
- Training: 10 complete passes through the sequence

Input		Output	
0110	*m*	0000	*a*
0000	*a*	0111	*n*
0111	*n*	1100	*y*
1100	*y*	1100	*y*
1100	*y*	0010	*e*
0010	*e*	0000	*a*
0000	*a*	1001	*r*
1001	*r*	1001	*s*
1001	*s*	0000	*a*
0000	*a*	0011	*g*
0011	*g*	0111	*o*

Predicting the Next Sound

- High error at the onset of words, but error decreases during a word



- High error at word onset demonstrates the network has discovered word boundaries

Remarks

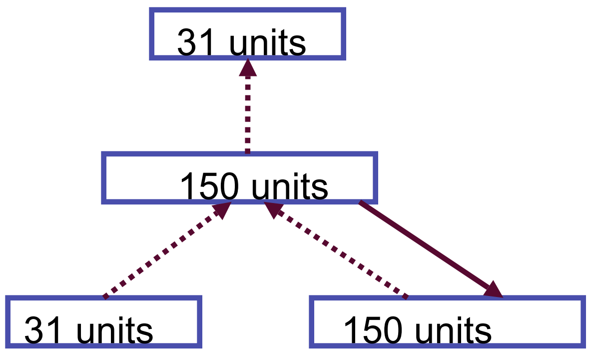
- Network learns statistics of co-occurrences
  - Criteria for boundaries is relative
  - Mistakes common compounds as individual words
    - Similar pattern in early child language acquisition
- Not a model of word acquisition
  - Listeners sometimes make ‘predictions’ from partial input, but it is not the main goal of language learning
  - Sound co-occurrences are only part of what identifies words

Discovering Lexical Classes

- Surface word order is influenced by many factors
  - Syntax, selectional and subcategorization restrictions, discourse factors …
  - Symbolic treatments appeal to relatively abstract, interacting rules which often depend on rich, hierarchical representations
- Can lexical classes be inferred from word order?
  - Verbs typically follow auxiliaries and precede determiners, nouns are often preceded by determiners
  - Also, selectional information: verbs are followed by specific kinds of nouns

Network Architecture

- Architecture:



- Input:
  - 1000 sentences of length 2-3,
    based on 29 words from 13 classes
  - Localist representation of each word (31 bits)
  - A sequence of 27,354 vectors

Input Structure

Categories of lexical items

Category	Examples
NOUN-HUM	man,woman
NOUN-ANIM	cat,mouse
NOUN-INANIM	book,rock
NOUN-AGRESS	dragon,monster
NOUN-FRAG	glass,plate
NOUN-FOOD	cookie,sandwich
VERB-INTRAN	think,sleep
VERB-TRAN	see,chase
VERB-AGPAT	move,break
VERB-PERCEPT	smell,see
VERB-DESTROY	break,smash

Template for sentence generator

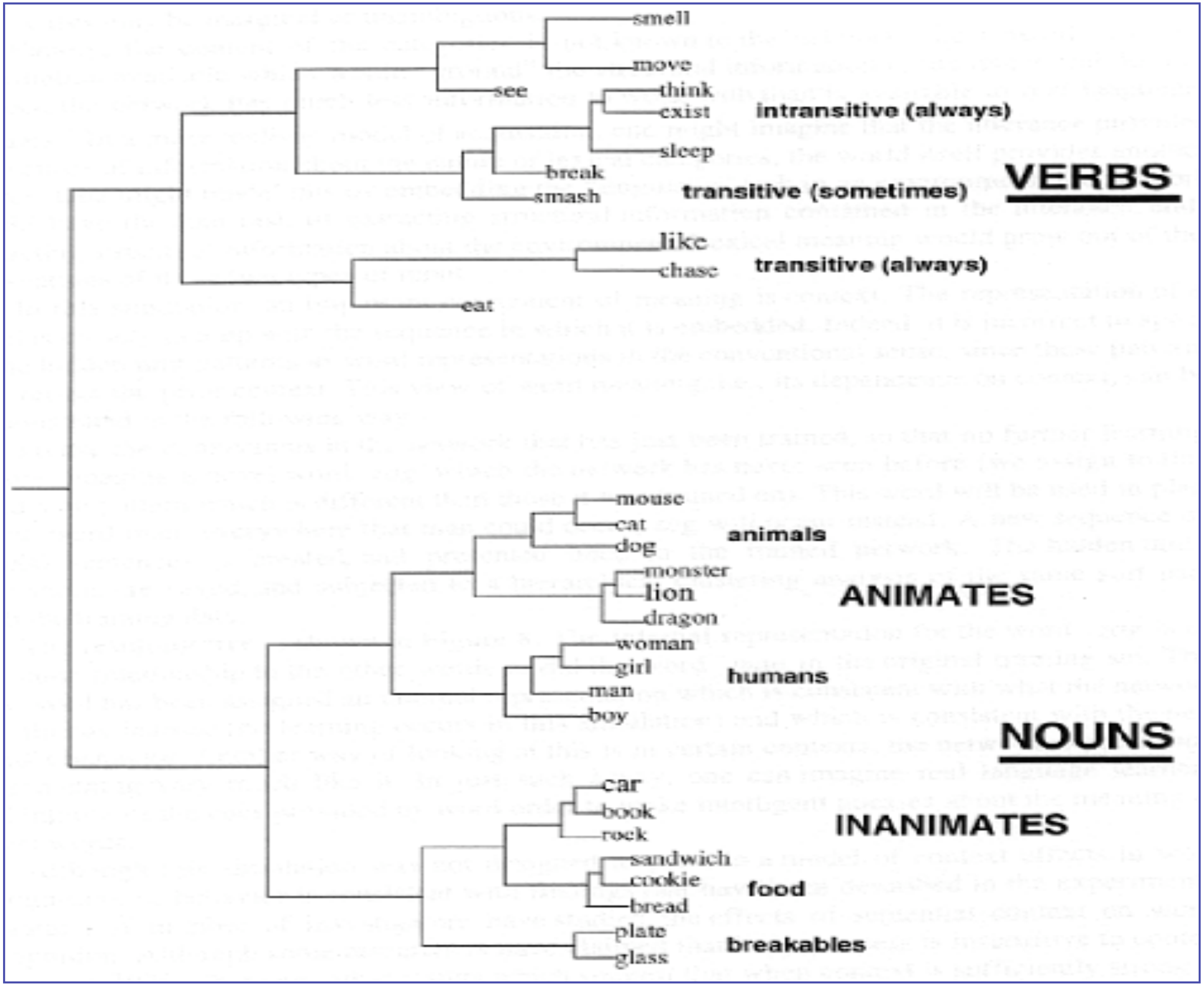
WORD 1	WORD 2	WORD 3
NOUN-HUM	VERB-EAT	NOUN-FOOD
NOUN-HUM	VERB-PERCEPT	NOUN-INANIM
NOUN-HUM	VERB-DESTROY	NOUN-FRAG
NOUN-HUM	VERB-INTRAN	
NOUN-HUM	VERB-TRAN	NOUN-HUM
NOUN-HUM	VERB-AGPAT	NOUN-ANIM
NOUN-HUM	VERB-AGPAT	
NOUN-ANIM	VERB-EAT	NOUN-FOOD
NOUN-ANIM	VERB-TRAN	NOUN-ANIM
NOUN-ANIM	VERB-AGPAT	NOUN-INANIM
NOUN-ANIM	VERB-AGPAT	

Predictions

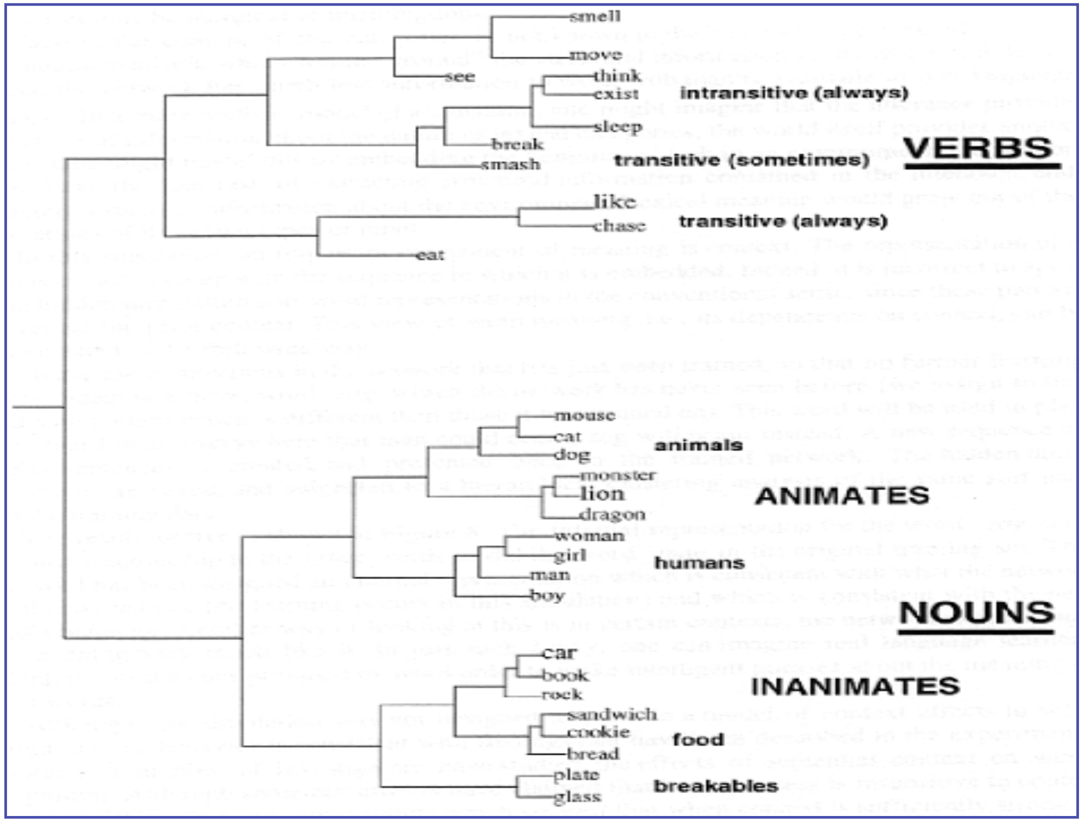
- Prediction is non-deterministic: next input is not random, but cannot be precisely predicted
  - Word order and selectional restrictions should partially constrain what words can appear next
  - The network should learn the frequency of occurrence of each possible successor
  - Output bit should be activated for all possible following words, with varying probability

Evaluation Procedure

- Compare network output to the probability vectors for each possible next word, given the current word and and context
  - Train the network on the input stream, and adjust weights
  - After training, for each word in a context, save the activation pattern of the hidden units as a vector
  - Hierarchically cluster the resulting vectors
- Lexical items with similar properties (i.e., contexts) are expected to be clustered together


smell
move
see
think
exist
intransitive (always)
sleep
break
smash
transitive (sometimes)
VERBS
like
chase
transitive (always)
eat
mouse
cat
animals
dog
monster
lion
ANIMATES
dragon
woman
girl
humans
man
boy
NOUNS
car
book
rock
INANIMATES
sandwich
cookie
food
bread
plate
breakables
glass

Cluster Analysis



- The network has discovered nouns vs. verbs, verb subcategorization, animates/inanimates, humans/animals, foods/breakables/objects...

Unknown Words

- In test data, replace *man* with a novel word *zog*
  - *Zog* is represented by a new input vector
  - *Zog* bears the same relationship to other words as *man* did in the original training set
- The new word's internal representation is based on its behaviour
- *Zog* is clustered in the exact same way as *man*

General discussion

- The network learns hierarchical lexical classes
  - Classes are inferred from word order / co-occurrence
  - Learning is purely based on observable data
    - No pre-specified localist representations, etc.
- Network predictions:
  - Context effects in processing: human lexical access is sensitive to context (e.g., Tabossi),
    - but there is evidence against immediate context effects in lexical access (e.g. Swinney)
  - Word classes are predicted, not individual words

Summary of SRNs …

- Finding structure in time/sequences:
  - Learns dependencies spanning over many transitions
  - Learns dependencies of variable length
  - Learns to make partial predictions from input
- Learning from various input encodings:
  - Structured: letter sequences where consonants have a distinguished feature
  - Random: words mapped to random 5 bit sequence
- Learns both general categories (types) and specific behaviours (tokens) based on context

Summary of Connectionist Models

- Connectionist models have appealing properties
    - Distributed computation and representation, single mechanism for the learning and use of knowledge
- But they have many limitations
    - Importance of starting small: more complex structures can only be learned after learning the simple ones
    - Scalability: they are not easily expandable to larger vocabularies and grammars
- Outstanding problems
    - Is grammatical structure really being learned?
    - Full linguistic complexity is hard to model (e.g., ambiguity, structural dependencies, ...)

Moving to Probabilistic Models

- Statistical/Probabilistic Models
  - Connectionist models have a highly probabilistic nature:
    - Learn regularities in a way which is sensitive to and reflect frequency
  - We can model language by directly applying probabilistic theory
  - We can combine symbolic and probabilistic approaches to achieve hybrid symbolic/sub-symbolic systems.