# Computational Psycholinguistics

# Lecture 13: Learning Linguistic Structure in Simple Recurrent Networks

Marshall R. Mayberry

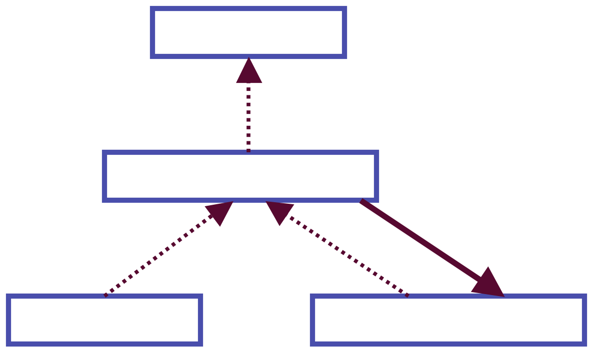
*Computerlinguistik*

*Universität des Saarlandes*

Reading: J Elman (1991). Distributed Representations, simple recurrent networks, and grammatical structure. *Machine Learning*.
J Elman (1993). Learning and development in neural networks: the importance of starting small. *Cognition*, **48**:71-99.

# SRNs

- Context units are <u>direct copies</u> of hidden units, the connections are not modifiable
  - Connections are one-to-one
  - Weights are fixed at 1.0
- Connections from context units to hidden units are modifiable; weights are learned just like all other connections
  - Training is done via the backpropagation learning algorithm



- Solution: let time be represented by its affect on processing
  - Dynamic properties which are responsive to temporal sequences
  - Memory

- Dynamical systems: "any system whose behaviour at one point in time depends in some way on its state at an earlier point in time"
  - See: *Rethinking Innateness*, Chapter 4.

# Structure of Training Environment

- 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 |
| VERB-EAT | eat |

- Template for sentence generator

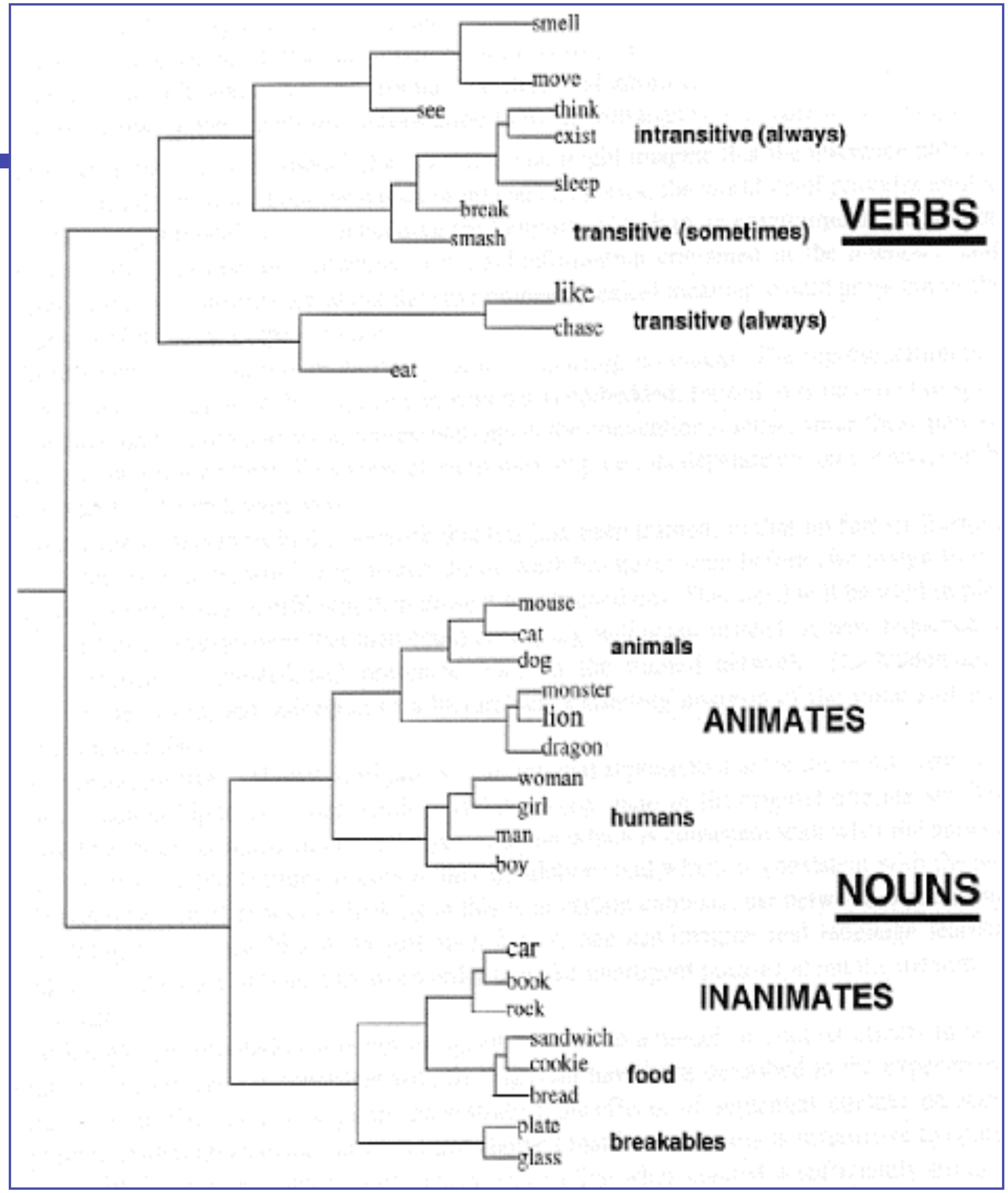
| WORD 1 | WORD 2 | WORD 3 |
|---|---|---|
| NOUN-HUM | VERB-EAT | NOUN-FOOD |
| NOUN-HUM | VERB-PERCEPT | NOUN-INANIM |
| NOUN-HUM | VERB-DESTROY | NOUN-FRAG |
| NOUN-HUM | VERB-INTRAN | |
| NOUN-HUM | VERB-TRAN | NOUN-HUM |
| NOUN-HUM | VERB-AGPAT | NOUN-ANIM |
| NOUN-HUM | VERB-AGPAT | |
| NOUN-ANIM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

# Calculating Performance

- Output should be compared to expected frequencies
- Frequencies are determined from the training corpus
  - Each word ($w_{input}$) in a sentence is compared with all other sentences that are up to that point identical (comparison set)
    - *Woman* *<u>smash</u>* *plate*
    - *Woman* *<u>smash</u>* *glass*
    - *Woman* *<u>smash</u>* *plate*
    - *…*
  - We then compute the vector of the probability of occurrence for each following word: this is the target, output for a particular input sequence
  - Vector: {0 0 0 p(plate|smash, woman) 0 0 p(glass|smash, woman) 0 … 0 }
  - This is compared to the output vector of the network, when the word *smash* is presented following the word *woman*.
- When performance is evaluated this way, RMS is 0.053
  - Mean cosine of the angle between output and probability: 0.916
    - This corrects for the fact that the probability vector will necessarily have a magnitude of 1, while the output activation vector need not.

# Cluster analysis:

- Lexical items with similar properties are grouped lower in the tree
- The network has discovered:
  - Nouns vs. Verbs
  - Verb subcategorization
  - Animates/inanimates
  - Humans/Animals
  - Foods/Breakables/Objects
- The network discovers ordering possibilities for various work categories and “subcategories”



# Type/Token distinction

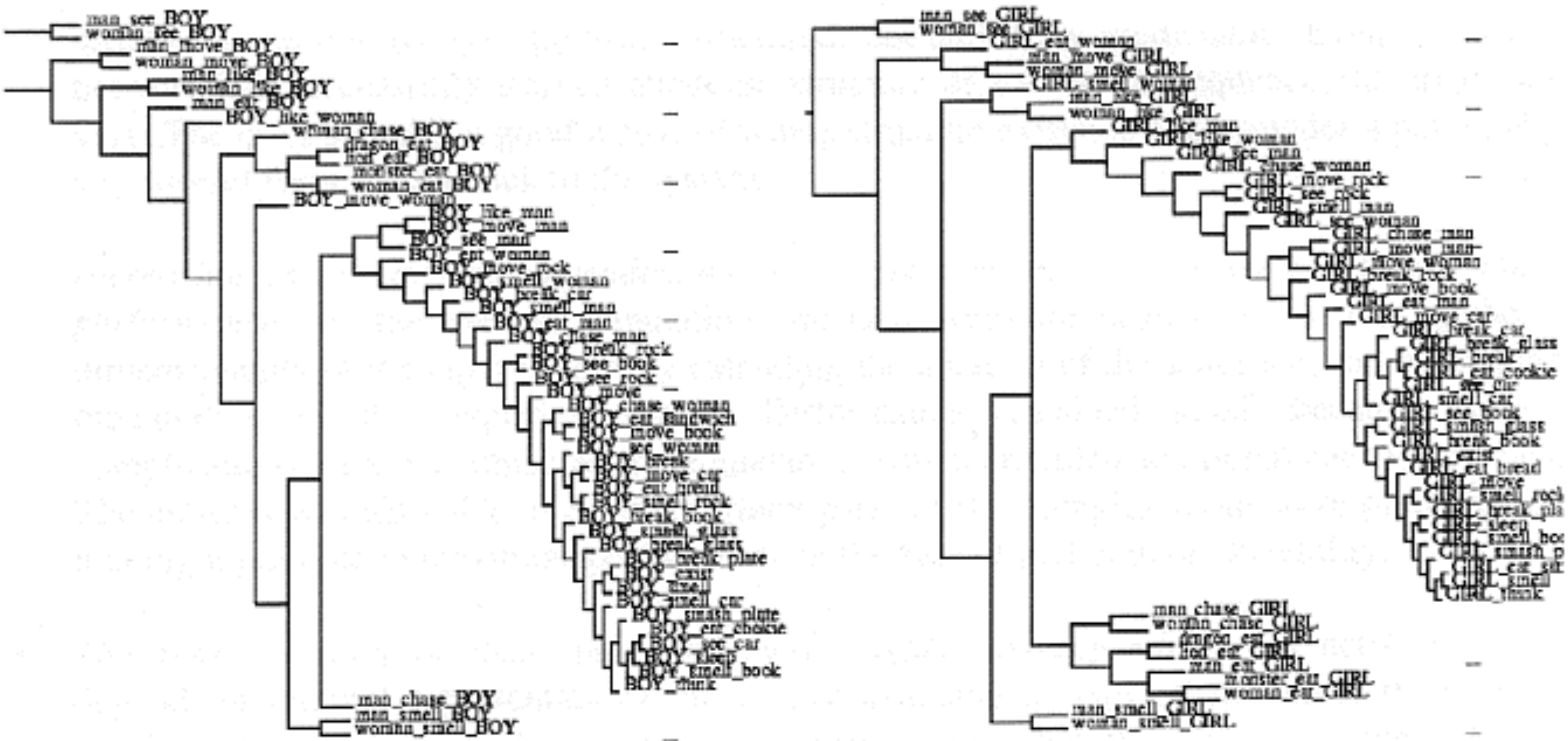
- Both symbolic systems and connectionist networks use representations to refer to things:
  - Symbolic systems use names
    - Symbols typically refer to well-defined classes or categories of entities
  - Networks use patterns of activations across hidden-units
    - Representations are highly context dependent
- The central role of context implies a distinct representation of *John*, for every context in with *John* occurs (which is an infinite number of $John_i$)
- Claim: distributed representations + context provides a solution to the representation of type/token differences
  - Distributed representations can learn new concepts as patterns of activation across a fixed number of hidden unit nodes
    - A fixed number of analog units can in principle learn an infinite number of concepts
  - Since SRN hidden units encode prior context, the hidden layer can in principle provide an infinite memory

# Type/Token continued

- In practice the number of concepts and memory is bounded
  - Units are not truly continuous (e.g. numeric precision on the computer)
  - Repeated application of logistic function to the memory results in exponential decay
  - Training environment may not be optimal for exploiting network capacity
  - Actual representational capacity remains an open question
- The sentence processing network developed representations reflecting aspects of the word's meaning and grammatical category
  - Apparent in the similarity structure of the "averaged" internal representation of each word: the network's representation of the word <u>types</u>
- The network also distinguishes between *specific* occurrences of words
  - The internal representation for each <u>token</u> of a word are very similar
  - But do subtly distinguish between the same word in different contexts
- Thus SRNs provide a potentially interesting account of the type-token distinction, which differs from the indexing or binding operations of symbolic systems.

# Clustering of word “tokens”

- Hierarchical clustering of specific occurrences of BOY and GIRL



# Summary of Elman 1990

- Some problems change their nature when expressed temporally:
  - E.g. sequential XOR developed frequency sensitive units
- Time varying error signal can be a clue to temporal structure:
  - Lower error in prediction suggests structure exists
- Increased sequential dependencies don't result in worse performance:
  - Longer, more variable sequences were successfully learned
  - Also, the network was able to make partial predictions (e.g. "consonant")
- The representation of time and memory is task dependent:
  - Networks intermix immediate task, with performing a task over time
  - No explicit representation of time: rather "processing in context"
  - Memory is bound up inextricably with the processing mechanisms
- Representation need not be flat, atomistic or unstructured:
  - Sequential inputs give rise to "hierarchical" internal representations

**"SRNs can discover rich representations implicit in many tasks, including structure which unfolds over time"**

# Challenges for a connectionist account

- What is the nature of the linguistic representations?
  - Localist representations seem too limited (fixed and simplistic)
  - Distributed are poorly understood, but greater capacity, can be learned
- How can complex structural relationships such as constituency be represented?  Consider "noun" versus "subject" versus "role":
  - The <u>boy</u> broke the *window*
  - The <u>rock</u> broke the *window*
  - The <u>*window*</u> broke
- How can the "open-ended" nature of language be accommodated by a fixed resource system?
  - Especially problematic for localist representations

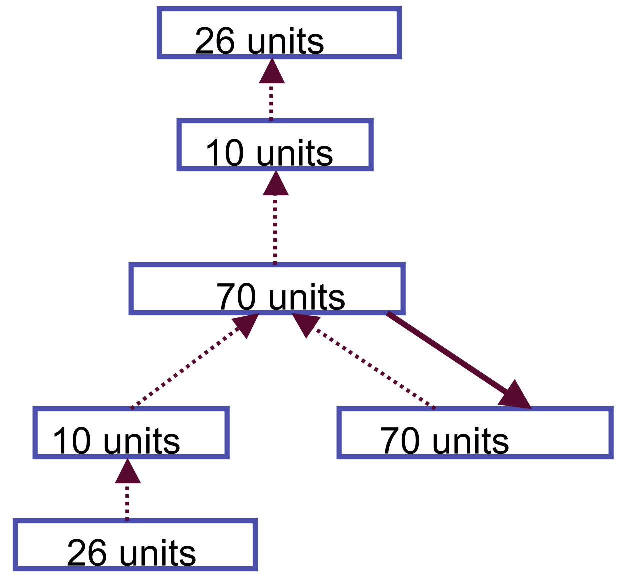
- In a famous article, Fodor & Pylyshyn argue that connectionist models:
  - Cannot encode for the fully compositional structure/nature of language
  - Cannot provide for the open-ended generative capacity

# Learning Linguistic Structure

- Construct a language, generated by a grammar which enforces diverse linguistic constraints:
  - Subcategorisation
  - Recursive embedding
  - Long-distance dependencies
- Training the network:
  - Prediction task
  - Structure of the training data is necessary
- Assess the performance:
  - Evaluation of predictions (as in Elman 1990), not RMS error
  - Cluster analysis? Only really informs us of the similarity of words, not the dynamics of processing
  - Principal component analysis: permits us to investigate the role of specific hidden units

# Learning Constituency: Elman (1991)

- So far, we have seen how SRNs can find structure in sequences
- How can complex structural relationships such as constituency be represented?
- The Stimuli:
  - Lexicon of 23 items
  - Encoded orthogonally, in 26 bit vector
- Grammar:
  - S ➔ NP VP "."
  - NP ➔ PropN | N | N RC
  - VP ➔ V (NP)
  - RC ➔ who NP VP | who VP (NP)
  - N ➔ boy | girl | cat | dog | boys | girls | cats | dogs
  - PropN ➔ John | Mary
  - V ➔ chase | feed | see | hear | walk |live | chases | feeds | sees | hears | walks | lives
  - Number agreement, verb argument patterns



# Training

- Verb subcategorization
  - Transitives: *hit, feed*
  - Optional transitives: *see, hear*
  - Intransitives: *walk, live*
- Interaction with relative clauses:
  - *Dog* *who chases cat* *sees girl*
  - *Dog* *who cat chases* *sees girl*
  - Agreement can span arbitrary distance
  - Subcategorization doesn't always hold (superficially)
- Recursion: Boys who girls who dogs chase see hear
- Viable sentences: where should end of sentence occur?
  - *Boys see (.) dogs (.) who see (.) girls (.) who hear (.) .*
- Words are not explicitly encoded for number, subcat, or category

# Training

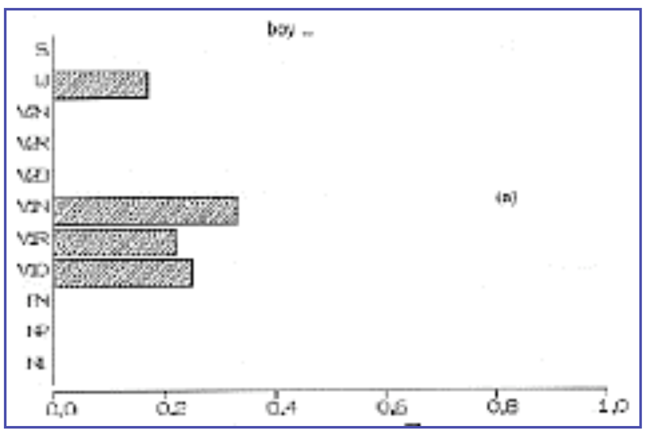
- At any given point, the training set contained 10000 sentences, which were presented to the network 5 times
- The composition of sentences varied over time:
  - Phase 1: Only simple sentences (no relative clauses)
    - 34,605 words forming 10000 sentences
  - Phase 2: 25% complex and 75% simple
    - Sentence length from 3-13 words, mean: 3.92
  - Phase 3: 50% complex, 50% simple, mean sentence length 4.38
  - Phase 4: 75% complex, 25% simple, max: 16, mean: 6

- WHY? Pilot simulations showed the network was unable to learn the task when given the full range of complex data from the beginning.
- Focussing on simpler data first, the network learned quickly, and was then able to learn the more complex patterns.
- Earlier simple learning, usefully constrained later learning

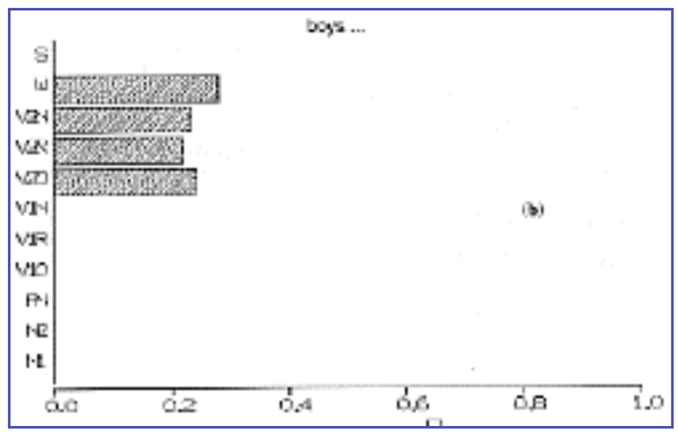
# Performance

- Weights are frozen and tested on a novel set of data (as in phase 4).
- Since the solution is non-deterministic, the network's outputs were compared to the context-dependent likelihood vector of all words following the current input (as done in the previous simulation)
  - Error was 0.177, mean cosine: 0.852
  - High level of performance in prediction
- Performance on specific inputs
- Simple agreement:

BOY ..

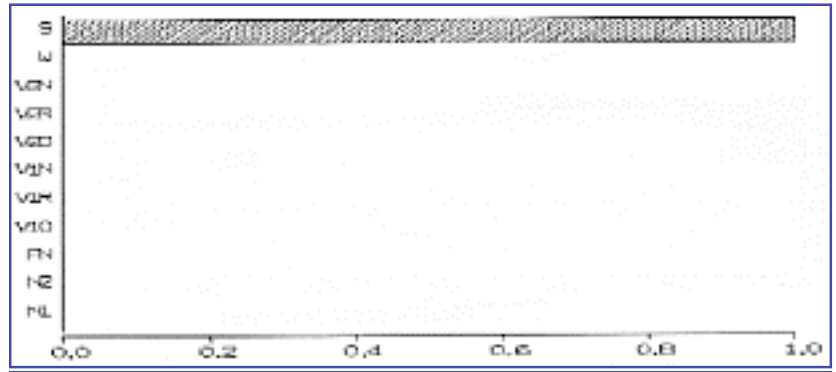



BOYS ..

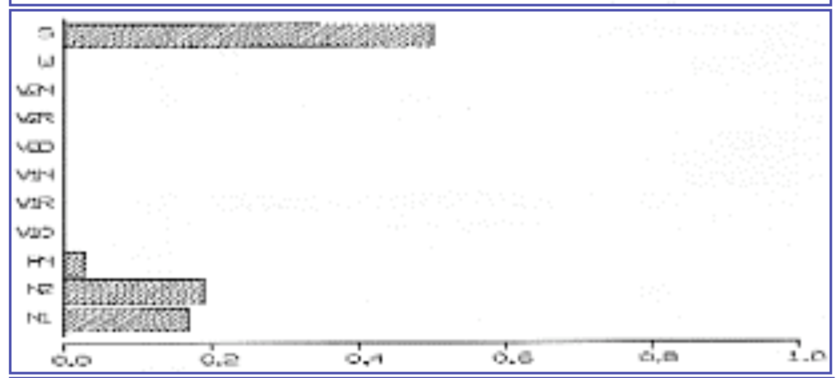


# Subcategorization

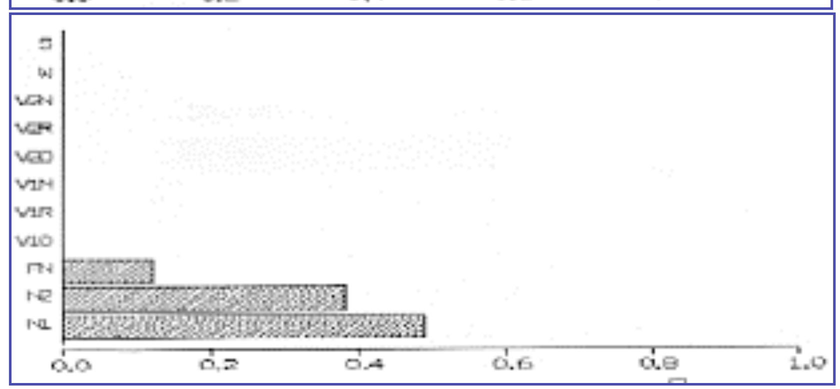
- Intransitive: “Boy lives …”
  - Must be a sentence, period expected




- Optional: “Boy sees …”
  - Can be followed by either a period,
  - Or some NP

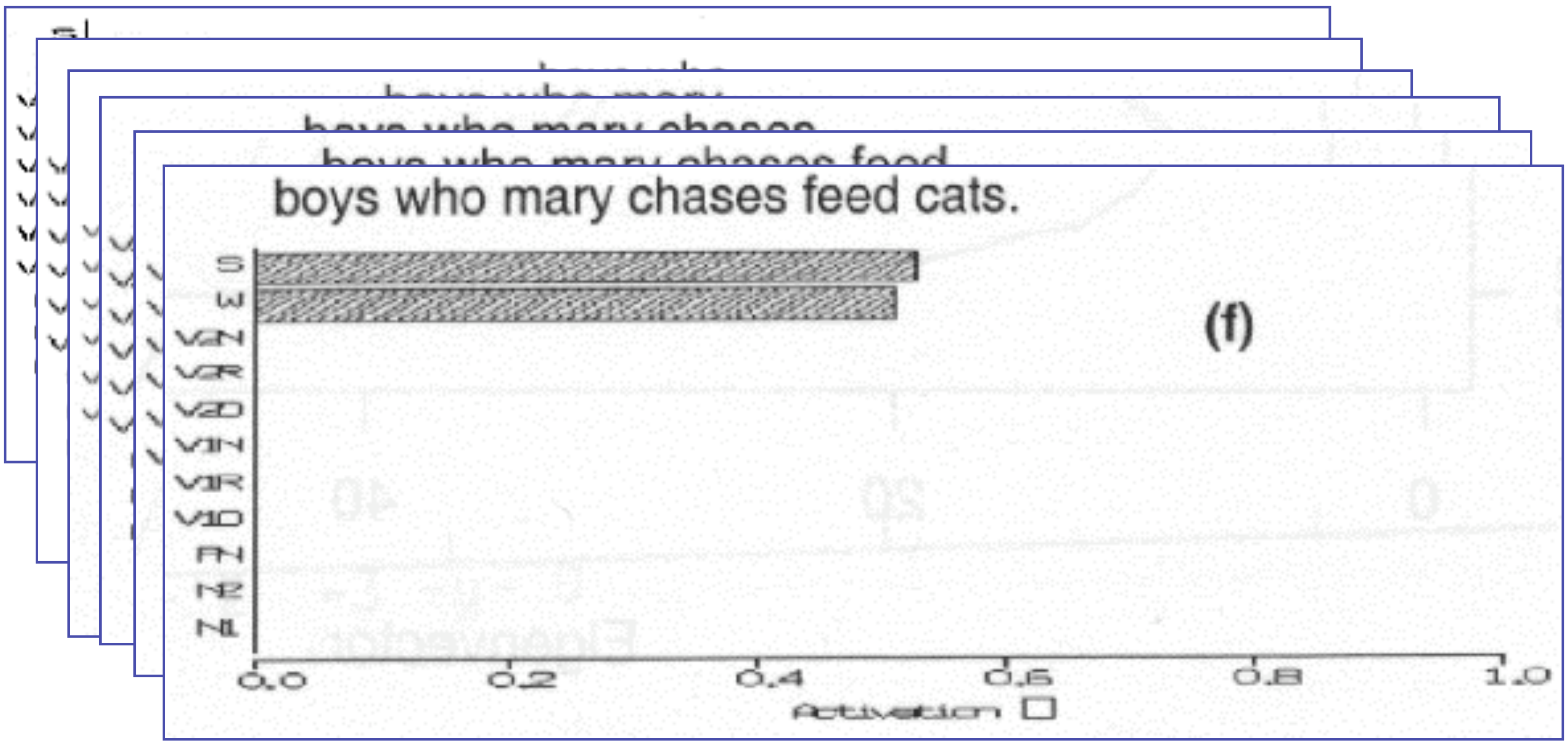



- Transitive: “Boy chases …”
  - Requires some object



# Processing complex sentences

- "boys who mary chases feed cats"
  - Long distance
    - Agreement: Boys … feed
    - Subcategorization: chases is transitive but in a relative clause
    - Sentence end: all outstanding "expectations" must be resolved



# Prediction reconsidered

- SRNs are trained on the *prediction* task:
  - “Self-supervised learning”:  no other teacher required
- Prediction forces the network to discover regularities in the temporal order of the input
- Validity of the the prediction tasks:
  - It is clearly not the “goal” of linguistic competence
  - But there is evidence that people can/do make predictions
  - Violated expectation results in distinct patterns of brain activity (ERPs)
- If children do make predictions, which are then falsified, this might constitute an indirect form of negative evidence, required for language learning.

# Results

- Learning was only possible when the network was forced to begin with simpler input
  - This effectively restricted the range of data to which the networks were exposed during initial learning
  - Contrasts with other results showing the entire dataset is necessary to avoid getting stuck in local minima (e.g. XOR)
- This behaviour partially resembles that of children:
  - Children do not begin by mastering language in all its complexity
  - They begin with simplest structures, incrementally building their “grammar”
- But the simulation achieves this by manipulating the environment:
  - This does not seem an accurate model of the situation in which children learn language
  - While adults do modify their speech, it is not clear they make such grammatical modifications
  - Children hear all exemplars of language from the beginning

# General results

- Limitations of the simulations/results:
  - Memory capacity remains un-probed
  - Generalisation is not really tested
    - Can the network inferentially extend what is known about the types of NPs learned to NPs with different structures
  - Truly a “toy” in terms of real linguistic complexity and subtlety
    - E.g. lexical ambiguity, verb-argument structures, structural complexity and constraints
- Successes
  - Representations are distributed, which means less rigid resource bounds
  - Context sensitivity, but can respond to contexts which are more “abstractly” defined
    - Thus can exhibit more general, abstract behaviour
    - Symbolic models are primarily context insensitive
- Connectionist models begin with local, context sensitive observations
- Symbolic models begin with generalisation and abstractions

# A Second Simulation

- While it's not the case that the environment changes, it's true that the child changes during the language acquisition period
- Solution: keep the environment constant, but allow the network to undergo change during learning
- Incremental memory:
  - Evidence of a gradual increase in memory and attention span in children
  - In the SRN, memory is supplied by the "context" units
  - Memory can be explicitly limited by depriving the network, periodically, access to this feedback
- In a second simulation, training began with limited memory span which was gradually increased:
  - Training began from the outset with the full "adult" language (which was previously unlearnable)

# Training with Incremental Memory

- Phase 1:
  - Training on corpus generated from the entire grammar
  - Recurrent feedback was eliminated after every 3 or 4 words, by setting all context units to 0.5
  - Longer training phase (12 epochs, rather than 5)
- Phase 2:
  - New corpus (to avoid memorization)
  - Memory window increased to 4-5 words
  - 5 epochs
- Phase 3: 5-6 word window
- Phase 4: 6-7 word window
- Phase 5: no explicit memory limitation implemented

- Performance: as good as on the previous simulation

# Analysing the solution

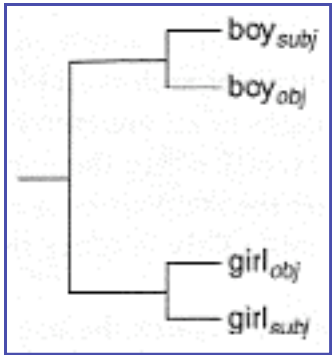
- Hidden units permit the network to derive a *functionally-based* representation, in contrast to a *form-based* representation of inputs

- Various dimensions of the internal representation were used for:
  - Individual words, category, number, grammatical role, level of embedding, and verb argument type
  - The high-dimensionality of the hidden unit vectors (70 in this simulation) makes direct inspection difficult

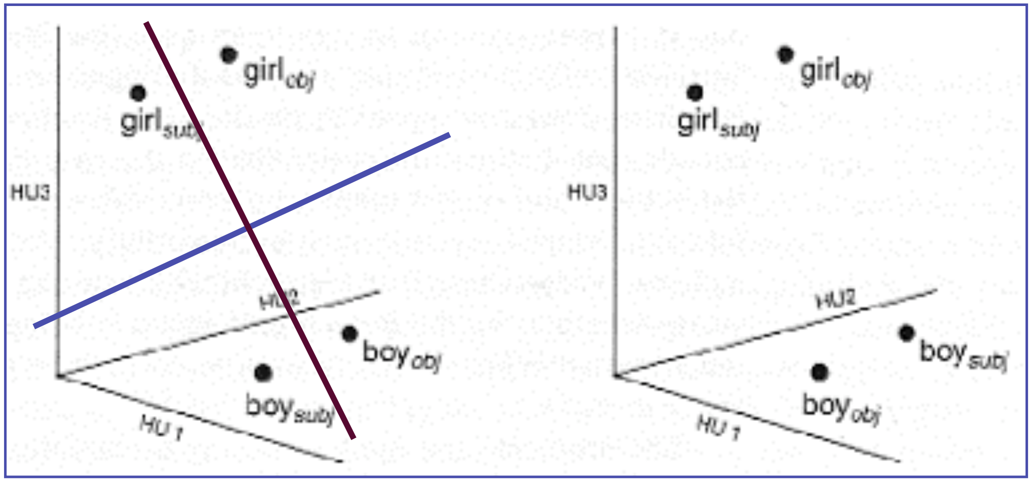
- Solution: Principal Component Analysis can be used to identify which dimensions of the internal state represent these different factors
  - This allows us to visualise the movement of the network through a state space for a particular factor, by discovering which units are relevant

# Principal Component Analysis

- Suppose we're interested in analysing a network with 3 hidden units and 4 patterns of activation, corresponding to: boy$_{subj}$, girl$_{subj}$, boy$_{obj}$, girl$_{obj}$
- Cluster analysis might reveal the following structure:
  - But nothing of the subj/obj representation is revealed




- If we look at the entire space, however, we can get more information about the representations:

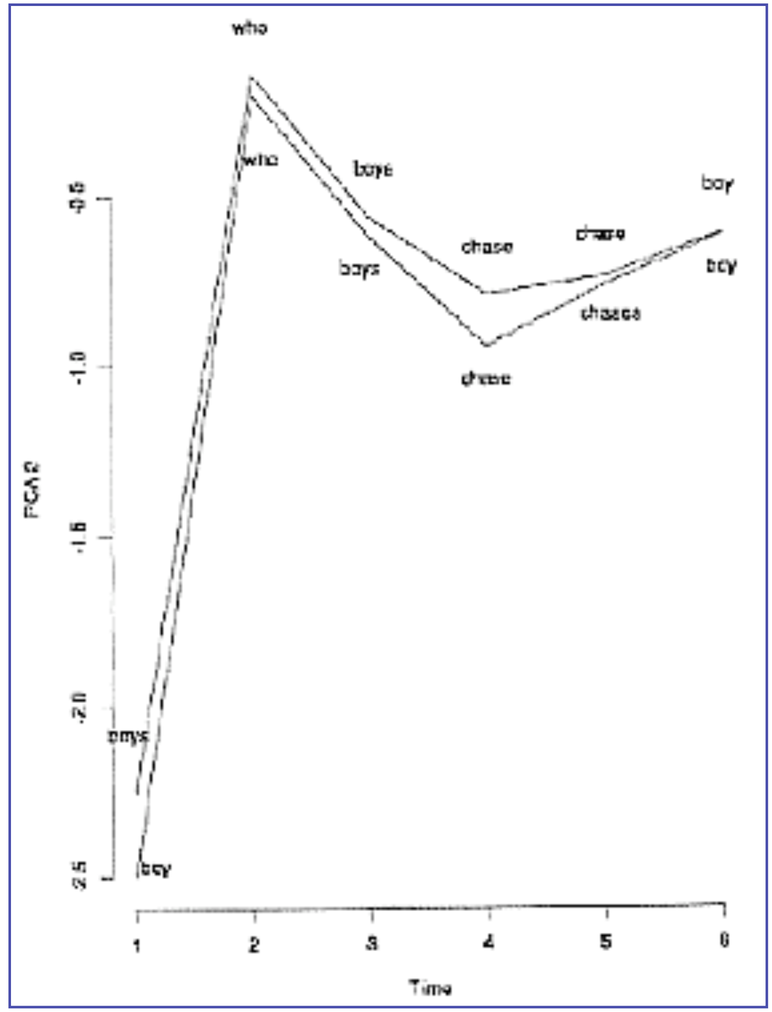



- Since visualising more than 3 dimensions is difficult, PCA permits us to identify which "units" account for most of the variation.
  - Reveals partially "localist" representations in the "distributed" hidden units

# Examples of Principal Components: 1

- Agreement
  - *Boy who boys chase chases boy*
  - *Boys who boys chase chase boy*

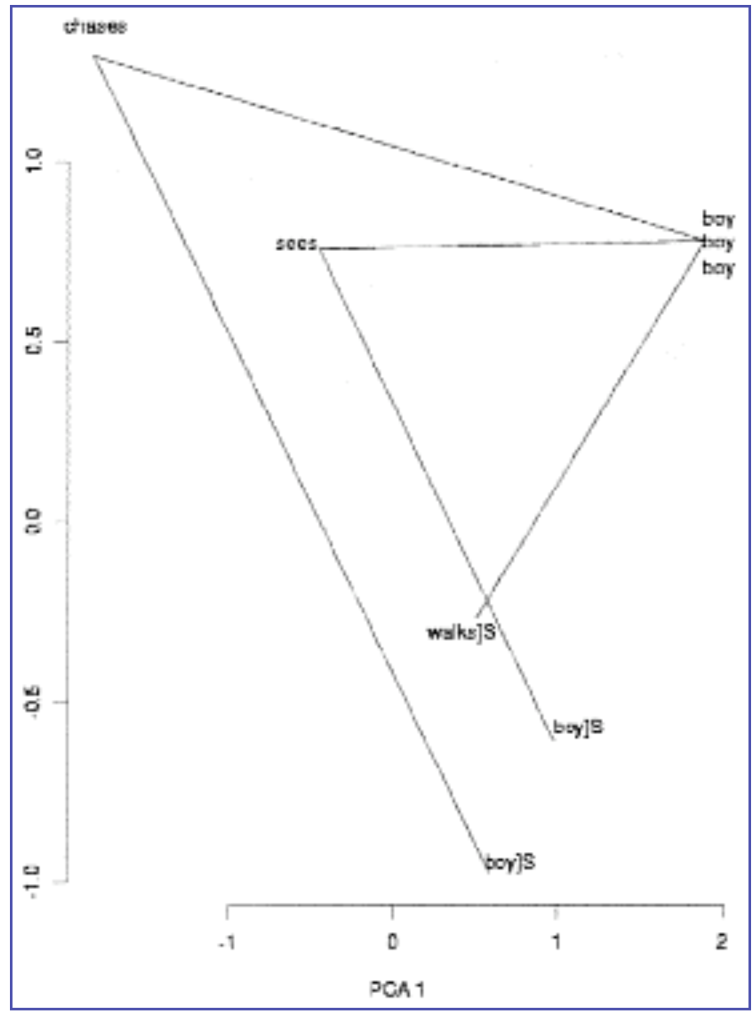
- The 2nd principal component encodes agreement in the main clause



# Examples of Principal Components: 2

- Transitivity
  - *Boy chases boy*
  - *Boy sees boy*
  - *Boy walks*

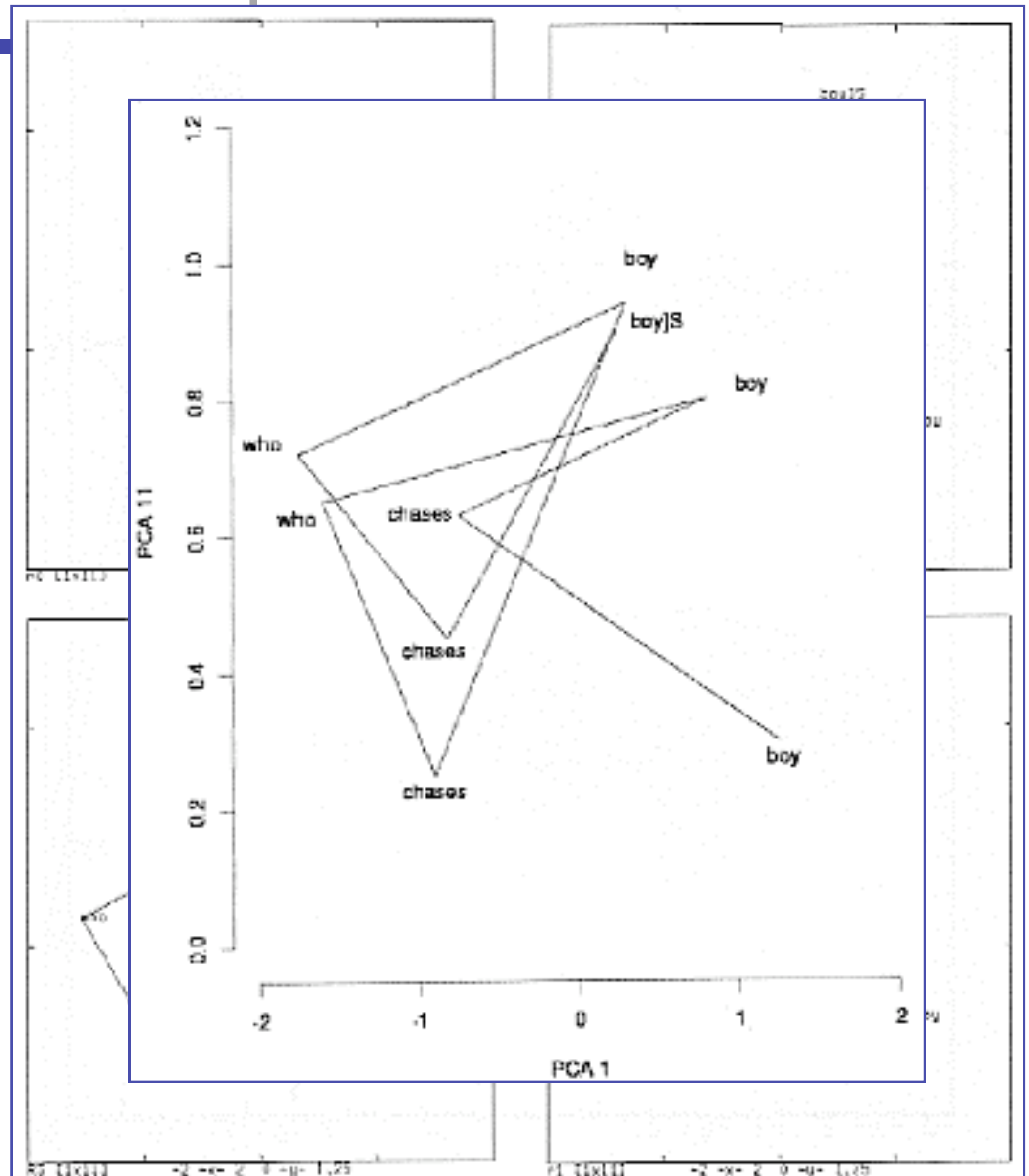
- Two principal components: 1 & 3
- PCA 1:
  - Nouns on the right
  - Verbs left
- PCA 3:
  - Intrans:  low
  - Optional trans:  mid
  - Transitive:  high



# Examples of Principal Components: 3

- Right embedding:
  - *Boy chases boy*
  - *Boy who chases boy chases boy*
  - *Boy chases boy who chases boy*
  - *Boy chases boy who chases boy who chases boy*

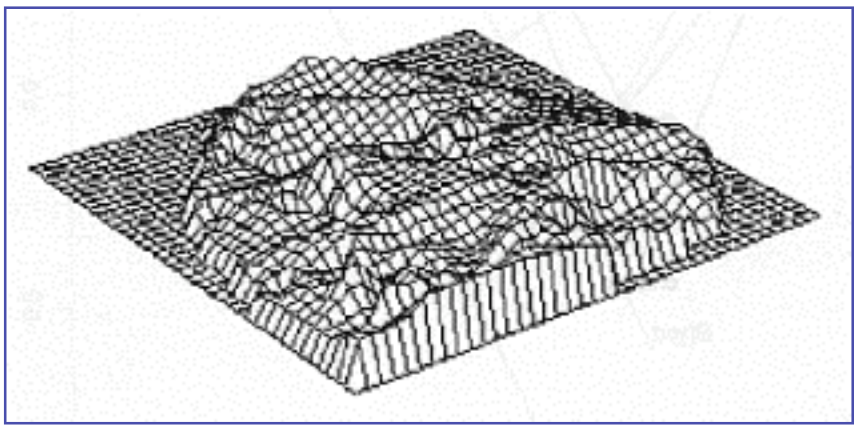
- PCA 11 and 1:
  - "Embedded clause are shifted to the left"
  - "RCs appear nearer the noun they modify"



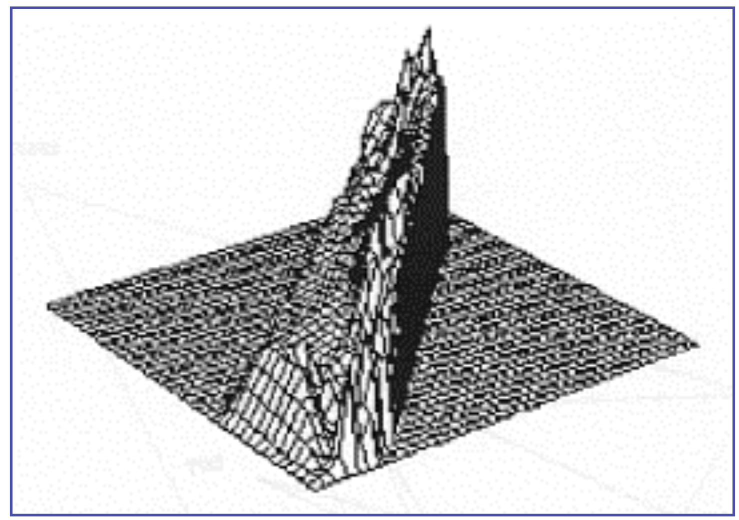
# PCA analysis of “Starting Small”

- We can use “Principal Component Analysis” to examine particularly important dimensions of the networks solutions more globally:
  - Sample of the points visited in the hidden unit space as the network processes 1000 random sentences
- The results of PCA after training:

Training on the full data set



Incremental training



The right plot reveals more clearly “organised” use of the state space

# Comments

- To solve the task, the network must learn the sources of variance (number, category, verb-type, and embedding)
- If the network is presented with the complete corpus from the start:
  - The complex interaction of these factors, long-distance dependencies, makes discovering the sources of variance difficult
  - The resulting solution is imperfect, and internal representation don’t reflect the true sources of variance
- When incremental learning takes place (in either form):
  - The network begins with exposure to only some of the data
    - Limited environment:  simple sentences only
    - Limited mechanisms:  simple sentences + noise (hence longer training)
  - Only the first 3 sources of variance, and no long-distance dependencies
- Subsequent learning is constrained (or guided) by the early learning of, and commitment to, these basic grammatical factors
  - Thus initial memory limitations permit the network to focus on learning the subset of facts which lay the foundation for future success

# The importance of starting small

- Networks rely on the representativeness of the training set:
  - Small samples may not provide sufficient evidence for generalisation
    - Possibly poor estimates of the population’s statistics
    - Some generalisations may be possible from a small sample, but are later ruled out
  - Early in training the sample is necessarily small
- The representation of experience:
  - Exemplar-based learning models store all prior experience, and such early data can then be re-accessed to subsequently help form new hypotheses
  - SRNs do not do this: each input has its relatively minor effect on changing the weights (towards a solution), and then disappears. Persistence is only in the change made to the network.
- Constraints on new hypotheses, and continuity of search:
  - Changes in a symbolic system may lead to suddenly different solutions
    - This is often ok, if it can be checked against the prior experience
  - Gradient descent learning makes it difficult for a network to make dramatic changes in its solution: search is continuous, along the error surface
  - Once committed to an erroneous generalisation, the network might not escape from a local minima

# Starting small (continued)

- Networks are most sensitive during the early period of learning:
  - Nonlinearity (the logistic activation function) means that weight modifications are less likely as learning progresses
    - Input is “squashed” to a value between 0 and 1
    - Nonlinearity means that the function is most sensitive for inputs around 0 (output is 0.5)
    - Nodes are typically initialised randomly about 0, so netinput is also near 0
    - Thus the network is highly sensitive
  - Sigmoid function become “saturated” for large +/- inputs
    - As learning proceeds units accrue activation
    - Weight change is a function of the error and slope of the activation function
    - This will become smaller as units’ activations become saturated, regardless of how large the error is
  - Thus escaping from local minima becomes increasingly difficult

- Thus, most learning occurs when information is least reliable

# Conclusions

- Learning language is difficult because:
  - Learning linguistic primitives is obscured by the full complexity of grammatical structure
  - Learning complex structure is difficult because the network lacks knowledge of the basic primitive representations
- Incremental learning shows how a system can learn a complex system by having better initial data:
  - Initially impoverished memory provides a natural filter for complex structures early in learning so the network can learn the basic forms of linguistic regularities
  - As the memory is expanded, the network can use what it knows to handle increasingly complex inputs
  - Noise, present in the early data, tends to keep the network in a state of flux, helping it to avoid committing to false generalisations

# Summary of SRNs …

- Finding structure in time/sequences:
  - Learns dependencies spanning more than a single transition
  - Learns dependencies of variable length
  - Learns to make partial predictions from structure input
    - Prediction of **consonants**, or particular lexical **classes**
- Learning from various input encodings:
  - Localist encoding: XOR and 1 bit per word
  - Distributed:
    - 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 purely on distributional evidence
- What are the limitations of SRNs?
  - Do they simply learn co-occurrences and contingent probabilities?
  - Can they learn more complex aspects of linguistic structure?

# Summary

- Implicit representation of time, reflected in the dynamic behaviour of the network:  not explicitly encoded.
- The importance of starting small:
  - Learning the more complex language was only possible by first learning simpler aspects of the grammar
- Outstanding problems:
  - Is grammatical structure really being learned?
  - Full linguistic complexity
    - Ambiguity:  lexical, syntactic, semantic
    - Structural:  subjacency, islands, extraction, …
    - Scale:  large lexicons, large structures
- 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.