Connectionist Symbol Processing

15-486/782: Artificial Neural Networks
David S. Touretzky

Fall 2006

1

What Is a Symbolic System?

- Symbols are arbitrary tokens.
- They combine to form structures.
- Rules govern derivation of new structures from old.

That's it!

Most of AI and CS is built on this framework.

Also, all of mathematics and logic.

2

Inference in Symbolic Systems

A grammar defines legal symbol structures.

A production rule in an expert system derives new working memory elements from old ones.

Logical deduction (Prolog, theorem provers).

Inductive reasoning (BACON).

Case-based reasoning.

Alpha-beta search.

All are part of the "symbolic paradigm."

3

What Do Neural Net Models Offer?

- Feature vector representations.
  - Dot product = similarity measure.
- A different approach to compositional structure.
- Different types of inference:
  - statistical learning (PCA, Boltzmann, Helmholtz, etc.)
  - nonlinear mapping (backprop)
  - parallel constraint satisfaction (Hopfield/Boltzmann)

A statistical, vector-based alternative to the symbolic framework. More like the brain???

4

Feature Vector Representations

The representation of “chair” should be more similar to “table” than to “banana”.

Shouldn't need axioms and rules to “deduce” this!

Ideally, the network should learn its own feature vector representations based on statistics.

See Hinton's “family trees” model on next slide.

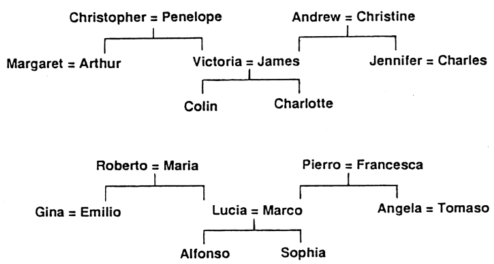
LSA (Latent Semantic Analysis):

Use SVD (like PCA) to derive “meaningful” feature vectors.

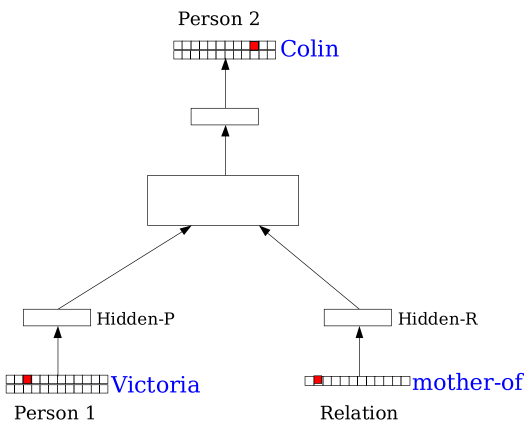
Hinton's Family Trees Example

English:

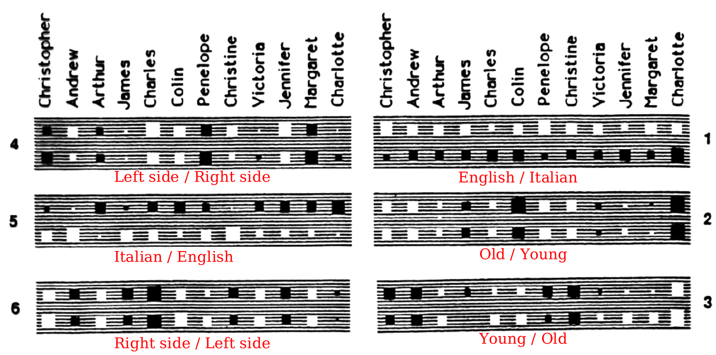
Italian:



Hinton's Family Trees Network



Hidden Layer Representation



Hinton's Family Trees: Generalization Performance

- Domain:
  - 24 people (12 English, 12 Italian)
  - 12 relations (mother, father, husband, wife, son, daughter, aunt uncle, brother, sister, nephew, niece)
  - 104 true relations over these family trees
- Training set:
  - Trained on 100 relations
- Test set:
  - Tested on 4 relations.
  - Got 3/4 right on one run, 4/4 right on another run.

Latent Semantic Analysis (LSA)

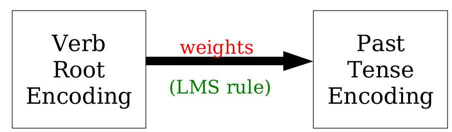
Method of constructing feature vectors for words.

- Build a frequency count table of words $\times$ contexts.
- A context can be a sentence, a paragraph, or a document.
- Apply semantic/contextual weighting functions.
- Perform SVD (Singular Value Decomposition).
- Use resulting values as feature vectors for words or documents.

Inference by Nonlinear Mapping

- Elman's SRN (Simple Recurrent Network) predicts next word in a sentence.
- NETtalk text to speech system learns complex mapping.
- Pollack's recurrent network learned regular grammars.
- McClelland & Rumelhart's past tense model.

McClelland & Rumelhart's Past Tense Model



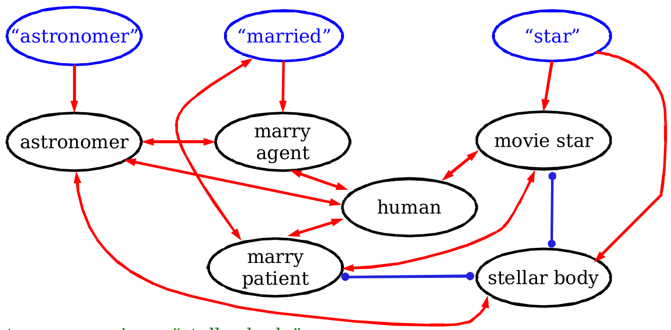
flip --> flipped
fib --> fibbed
fit --> fitted
Regular forms

go --> went
sing --> sang
buy --> bought
Irregular forms

bring --> brought / brang / broughted

Parallel Constraint Satisfaction

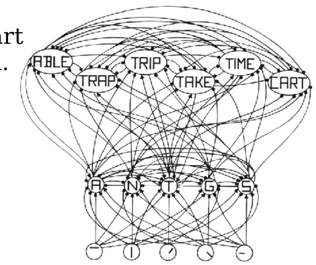
"The astronomer married the star." (Pollack & Waltz)



"Astronomer primes "stellar body",
but"movie star" eventually wins.

More Parallel Constraint Satisfaction

- Hopfield nets for Traveling Salesman Problem
- Boltzmann machines
- Recurrent backprop nets that learn attractor states
- McClelland & Rumelhart word perception model.



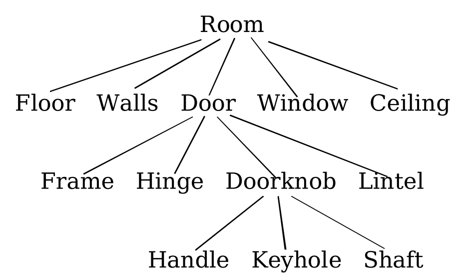
Problems of Compositional Structure

- Structures are (in principal) unbounded, but vectors have finite length.
- Is dot product still a meaningful similarity measure?
- How can we get the systematicity that comes so easily to symbolic systems?
- Can we learn composite representations?
- Construct new structures on the fly?

Conflict?

Reduced Descriptions (Hinton)

"Pointer following is expensive, so avoid it."



Does a room have a keyhole? Does a cat have nostrils?

Holographic Reduced Representations (Tony Plate)

- Use large (N = 2000 elements) vectors.
  - Elements drawn independently from Normal(0, 1/N)
  - Vectors have roughly zero mean.
  - No similarity: $a \cdot b \approx 0$
  - Not a feature vector representation.
- Combine vectors using circular convolution operator.

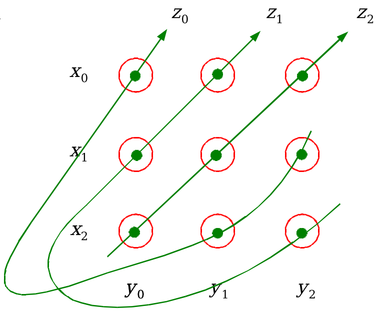
$$C = a \times b$$

- Sort of like "cons" in Lisp: associates **a** with **b**.

Circular Convolution Operator

$$z = x \times y$$

$$z_i = \sum_{k=0}^{N-1} x_k y_{(i-k) mod N}$$



Circular Convolution Properties

Commutative	$a \times b = b \times a$
Associative	$a \times (b \times c) = (a \times b) \times c$
Bilinear	$a \times (\beta b + \gamma c) = \beta a \times b + \gamma a \times c$
Identity	$I \times a = a$
Zero	$\bar{0} \times a = \bar{0}$
Inverse	$a^{-1} \times a = I$

Inverse of a Vector

A true inverse a^{-1} exists under certain conditions, so that $a^{-1} \times a = I$, but it may be numerically unstable.

If x is drawn from Normal(0,1/N) then it has an ***approximate*** *inverse $x_i^T = x_{-i(mod N)}$*

Taking Apart a Pair

Given: $a \times b$
To get back b, multiply by a^T:

$$a^T \times (a \times b) \;=\; (a^T \times a) \times b \;\approx\; I \times b \;=\; b$$

Since a^T *only approximates the inverse, we really get*

$$\hat{b} \;=\; b + \eta .$$

Use a separate 'cleanup' process to eliminate noise:

$$b + \eta \;\rightarrow\; b$$

How to do cleanup?
Associative memory
Nearest-neighbor match

Encoding Propositions

Nouns: *John, Mary, Spot*
Verbs: *bite, kiss*
Roles: $bite_{agt}$, $bite_{obj}$, $kiss_{agt}$, ...

'Spot bit Mary'

$$C \;=\; bite \;+\; bite_{agt} \times Spot \;+\; bite_{obj} \times Mary$$

Who was the biter? $C \times bite_{agt}^T \;\approx\; Spot \;+\; noise$
Who was bitten? $C \times bite_{obj}^T \;\approx\; Mary \;+\; noise$
What was the action? $C \;\approx\; bite \;+\; noise$
What did Spot do? $C \times spot^T \;\approx\; bite_{agt}$

Convolution Preserves Similarity

$$sim(a \times b, c \times d) \;\sim\; sim(a, c) \cdot sim(b, d)$$

Superpositions are similar to each of their components, to a decreasing degree as more patterns are added.

$$sim(a, a+b) \;>\; sim(a, a+b+c) \;>\; sim(a, b)$$

Similarity can be measured by dot product.

Normalization

All vectors should be normalized to unit length:

$$z \;=\; x \times y$$

$$\langle z \rangle \;=\; \frac{z}{\|z\|}$$

Advantages:

- unbiased dot product similarity scores
- equal weighting of all component patterns
- reduced noise in decoded slot fillers

Feature Representations

$$\begin{aligned} John &= \langle animal + human + id_{John} \rangle \\ Jane &= \langle animal + human + id_{Jane} \rangle \\ Spot &= \langle animal + dog + id_{Spot} \rangle \end{aligned}$$

John is more similar to Jane than to Spot.

Recursive Composition

'Spot bit Jane, causing Jane to flee from Spot.'

$$P_{bite} = \langle bite + \langle spot + jane \rangle + bite_{agt} \times spot + bite_{obj} \times jane \rangle$$

$$P_{flee} = \langle flee + \langle spot + jane \rangle + flee_{agt} \times jane + flee_{from} \times spot \rangle$$

$$P = \langle cause + \langle P_{bite} + P_{flee} \rangle + cause_{antc} \times P_{bite} + cause_{cnsq} \times P_{flee} \rangle$$

P contains a 'reduced description' of Jane being bitten.

Analogy Retrieval

Describe a situation in terms of predicates and objects.

$$\text{'Spot bit Fred'} = bit(Spot, Fred)$$

Find an 'analogous' situation:

Same predicate? $bit(Fido, John)$
Same objects? $saw(Fred, Spot)$
Same predicate & objects? $bit(Fred, Spot)$
Same object-roles? $bit(Spot, Jane)$

Analogy Retrieval

- Ken Forbus' MAC/FAC is a symbolic analogy-making program.
- Use <u>content vectors</u> to summarize a description by counting the # of occurrences of each object and predicate.
- Dot product of content vectors heuristically estimates similarity.
- High scoring descriptions are then explored symbolically to verify the analogy.
- But content vectors ignore structural similarity: “John bit Fido” treated the same as “Fido bit John”.

A Better Retrieval Heuristic

Holographic Reduced Representations (HRRs)
encode structural similarity:

$$\langle bite + \langle John + Fido \rangle + bite_{agt} \times John + bite_{obj} \times Fido \rangle$$

Structural Similarity

People are sensitive to structural similarity.

"Spot bit Jane, causing Jane to flee from Spot."

LS (Literal Similarity)

Fido bit John, causing John to flee from Fido.

SF (Surface Features)

John fled from Fido, causing Fido to bite John.

CM (cross-mapped analogy) – roles switched

Fred bit Rover, causing Rover to flee from Fred.

Structural Similarity (cont.)

AN (analogy)

Mort bit Felix, causing Felix to flee from Mort.

a mouse

a cat

FOR (first-order relations only)

Mort fled from Felix, causing Felix to bite Mort.

How Humans Handle Similarity

For LTM retrieval in humans:

$$LS > CM \geqslant SF > AN \geqslant FOR$$

		Commonalities with probe			Similarity scores	
P	Spot bit Jane, causing Jane to flee from Spot.					
	Episodes in long-term memory:	Object attributes	First-order relation names	Higher-order structure	HRR	MAC
$\mathbf{E_{LS}}$	Fido bit John, causing John to flee from Fido.	✓	✓	✓	0.71	1.0
$\mathbf{E_{SF}}$	John fled from Fido, causing Fido to bite John	✓	✓	×	0.47	1.0
$\mathbf{E_{CM}}$	Fred bit Rover, causing Rover to flee from Fred.	✓	✓	✓	0.47	1.0
$\mathbf{E_{AN}}$	Mort bit Felix, causing Felix to flee from Mort.	×	✓	✓	0.42	0.6
$\mathbf{E_{FOR}}$	Mort fled from Felix, causing Felix to bite Mort.	×	✓	×	0.30	0.6

Summary

- Connectionist symbol processing proposes a statistical, vector-based alternative to the "classic" symbolic paradigm.
- Issues:
  - feature vector encoding
  - compositionality (difficult)
  - nonlinear mapping
  - parallel constraint satisfaction

  Representation

  Inference
- Interesting ideas in all four areas.
- No one has figured out how to combine all four. Still in the early days of this research.