Sentence Processing in a Vectorial Model of Working Memory

William Schuler
Department of Linguistics,
The Ohio State University

June 29, 2014

Introduction

I'm envious of my computational cog neuro colleagues; they define. . .

- associative memory in terms of neural activation (vector prod. model).

  [Marr, 1971, Anderson et al., 1977, Murdock, 1982, McClelland et al., 1995, Howard and Kahana, 2002]

  - one (possibly superposed) activation-based state: cortex as vector
  - a set of weight-based cued associations: hippocampus as matrix
- neural activation in terms of ligands, receptors, chemistry, physics.

I'd like to define parsing in terms of (vectorial) associative memory models!

But existing sent. proc. models don't do parsing / connect to vector memory:

- connectionist models don't explain why syntactic prob. is so predictive. (subjacency, gap propagation to modifiers, . . .)

  [Fossum and Levy, 2012, van Schijndel et al., 2013b, van Schijndel et al., 2014]
- ACT-R is a good candidate, but it is serial (ditto GP, construal, race). vector state can easily be superposed, why not in sentence proc?
- full parallel surprisal accounts don't explain center embedding effects. superposing distinct analyses requires huge tensors, then all available.

Introduction

So I'll build a model based on our earlier symbolic parallel model:

- builds 'incomplete categories' in left-corner parse [Schuler et al., 2010]:
  - top-down for right children, to build 'awaited' category: S/VP V → S/NP
  - bottom-up for left children, to build 'active' category: NP/N N → S/VP
- unlike earlier work, syntactic category states are superposed in vector
- constraints on 'awaited' categories are multiplied in at right children
- constraints on 'active' categories are reconstructed at left children

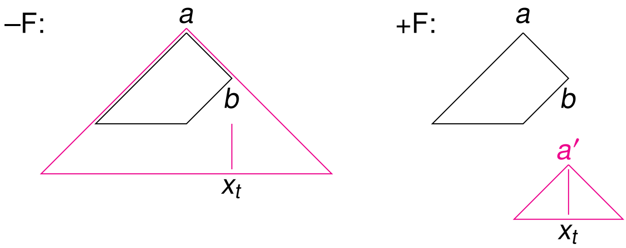
Results:

- seems to work, theoretically justifies parallel left-corner parsing model
- predicts processing difficulty in center embedding:
  - result of noise in reconstruction after multiplied-in constraints

(Warning: 'existence proof' results, not a state-of-the-art parser.)

Previous Work: Left-corner Parsing

In left-corner parse [van Schijndel et al., 2013a], either do a fork or don't:



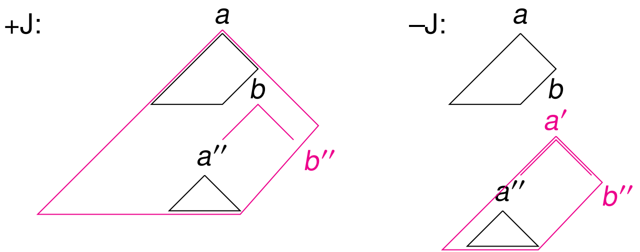
Build a complete category (triangle).

$$\frac{a/b \quad x_t}{a}\; b \rightarrow x_t \qquad \text{(–F)}$$

$$\frac{a/b \quad x_t}{a/b \quad a'}\; b \xrightarrow{+} a' \,...\, ; \quad a' \rightarrow x_t \qquad \text{(+F)}$$

Previous Work: Left-corner Parsing

Then, either do a join or don't (incrementally build top-down or bottom-up):



Build incomplete category (trapezoid) out of complete category (triangle).

$$\frac{a/b \quad a''}{a/b''}\; b \rightarrow a''\; b'' \qquad (+\mathrm{J})$$

$$\frac{a/b \quad a''}{a/b \quad a'/b''}\; b \xrightarrow{+} a' \;...\;; \quad a' \rightarrow a''\; b'' \qquad (–\mathrm{J})$$

Previous Work: Vectorial Memory

Model connections in associative memory w. matrix [Anderson et al., 1977]:

$$v = M\,u \tag{1}$$

$$(M\,u)_{[i]} \stackrel{\text{def}}{=} \textstyle\sum_{j=1}^{J} M_{[i,j]} \cdot u_{[j]} \tag{1$'$}$$

Build cued associations using outer product:

$$M_t = M_{t\text{–}1} + v \otimes u \tag{2}$$

$$(v \otimes u)_{[i,j]} \stackrel{\text{def}}{=} v_{[i]} \cdot u_{[j]} \tag{2$'$}$$

Combine cued associations using pointwise / diagonal product:

$$w = \text{diag}(u)\,v \tag{3}$$

$$(\text{diag}(v)\,u)_{[i]} \stackrel{\text{def}}{=} v_{[i]} \cdot u_{[i]} \tag{3$'$}$$

Vectorial Parser

We can implement the two left-corner parser phases using these operations.

Here's what we need:

Permanent 'procedural' associations (separate matrices, for simplicity):

- associative store for preterminal category given observation:
  $P = \sum_i p_i \otimes x_i$
- associative store for grammar rule given parent / l. child / r. child:
  $G = \sum_i g_i \otimes c_i; \quad G' = \sum_i g_i \otimes c'_i; \quad G'' = \sum_i g_i \otimes c''_i$
- associative store for l. descendant category given ancestor category:
  $D'_0 \leftarrow \mathrm{diag}(\mathbf{1}); \quad D_0 \leftarrow \mathrm{diag}(\mathbf{0}); \quad D'_k \leftarrow G'^{\top} G\, D'_{k-1}; \quad D_k \xleftarrow{+} D'_{k-1}$
- associative store for r. descendant category given ancestor category:
  $E'_0 \leftarrow \mathrm{diag}(\mathbf{1}); \quad E_0 \leftarrow \mathrm{diag}(\mathbf{0}); \quad E'_k \leftarrow G''^{\top} G\, E'_{k-1}; \quad E_k \xleftarrow{+} E'_{k-1}$

Vectorial Parser

We'll also need:

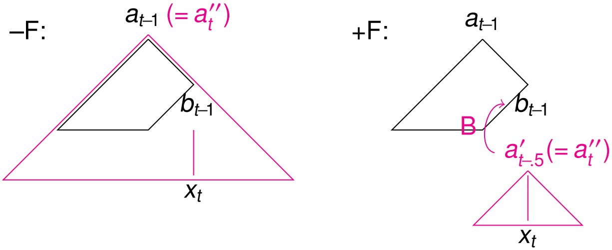
Temporary state vector 'working memory':

- lowest awaited node: b (can be superposed, of course)
- observations: x (word token)

Temporary associations (separate matrices, for simplicity):

- associative store for 'active' node above 'awaited' node: A
- associative store for 'awaited' node above 'active' node: B
- associative store for category type of node: C

Vectorial Parser - ‘fork’ phase



$c_t^- = \text{diag}(P\,x_t)\;C_{t\text{–}1}\;b_{t\text{–}1}$ (no-fork preterminal category combines x, b)

$c_t^+ = \text{diag}(P\,x_t)\;D\;C_{t\text{–}1}\;b_{t\text{–}1}$ (forked preterminal category goes through D)

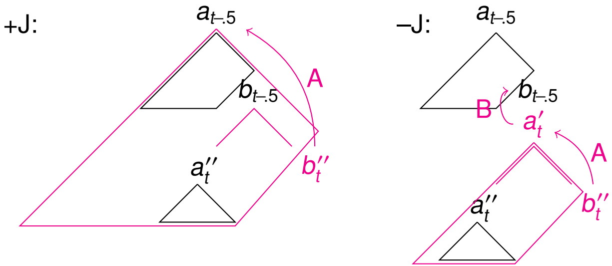
$a_{t\text{–}.5},\, a'_{t\text{–}.5} \sim \text{Exp}$ (100 of $10^{\mathbb{R}^{20}_{-150}}$ to be sparse, avoid over-/underflow)

$a_{t\text{–}1} = A_{t\text{–}1}\;b_{t\text{–}1}$ (define a)

$B_{t\text{–}.5} = B_{t\text{–}1} + b_{t\text{–}1} \otimes a'_{t\text{–}.5} + B_{t\text{–}1}\;a_{t\text{–}1} \otimes a_{t\text{–}.5}$ (update B for new nodes)

$C_{t\text{–}.5} = C_{t\text{–}1} + c_t^+ \otimes a'_{t\text{–}.5} + \text{diag}(C_{t\text{–}1}\;a_{t\text{–}1})\;E^\top c_t^- \otimes a_{t\text{–}.5}$ (reconstruct via E)

Vectorial Parser - 'join' phase



$$g^+_t = \mathrm{diag}(G'\, C_{t-.5}\, a''_t)\, G\, C_{t-.5}\, b_{t-.5}$$ (join rule combines categories of a'', b)

$$g^-_t = \mathrm{diag}(G'\, C_{t-.5}\, a''_t)\, G\, D\, C_{t-.5}\, b_{t-.5}$$ (no-join rule goes through D)

$$a'_t, b''_t \sim \mathrm{Exp}$$ (100 of $10^{\mathbb{R}^{20}_{-150}}$ to be sparse, avoid over-/underflow)

$$A_t = A_{t-1} + \frac{A_{t-1}\, b_{t-.5}\, \|g^+_t\| + a'_t\, \|g^-_t\|}{\|A_{t-1}\, b_{t-.5}\, \|g^+_t\| + a'_t\, \|g^-_t\|\|} \otimes b''_t$$ (update A w. weighted avg)

$$B_t = B_{t-.5} + b_{t-.5} \otimes a'_t$$ (define B for a')

$$C_t = C_{t-.5} + G^\top g^-_t \otimes a'_t + \frac{G''^\top g^+_t + G''^\top g^-_t}{\|G''^\top g^+_t + G''^\top g^-_t\|} \otimes b''_t$$ (update C w. weighted avg)

Vectorial Grammar

Parser accepts PCFGs: (note this grammar can be center-embedded)

$$
\begin{aligned}
P(T \rightarrow S\ T) &= 1.0 \\
P(S \rightarrow NP\ VP) &= 0.5 \\
P(S \rightarrow IF\ S\ THEN\ S) &= 0.25 \\
P(S \rightarrow EITHER\ S\ OR\ S) &= 0.25 \\
P(IF \rightarrow if) &= 1.0 \\
P(THEN \rightarrow then) &= 1.0 \\
P(EITHER \rightarrow either) &= 1.0 \\
P(OR \rightarrow or) &= 1.0 \\
P(NP \rightarrow kim) &= 0.5 \\
P(NP \rightarrow pat) &= 0.5 \\
P(VP \rightarrow leaves) &= 0.5 \\
P(VP \rightarrow stays) &= 0.5
\end{aligned}
$$

Predictions

This parser can process short sentences using a simple associative store (meaning it usually predicts a top-level category at the correct position):

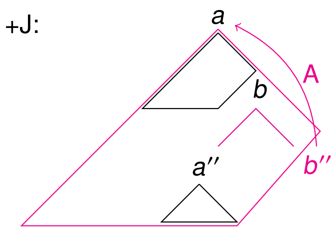
condition	correct	incorrect
right-branching:		
If Kim stays then if Kim leaves then Pat leaves.	297	203
center-embedded:		
If either Kim stays or Kim leaves then Pat leaves.	231*	269

And it also predicts difficulty at center embedded constructions (*$p < .001$)!

Predictions

Why is center embedding difficult for this model?

- traversal to r. child multiplies constraints on *b*, eliminates hypotheses. e.g. if *b* is S or NP (say after *know*), then after word *the*, b'' must be N.



- traversal from l. child reconstructs constraints on *a* using b'', but lossy. e.g. if *a* was S or NP, after *the dog*: b'' is N, reconstructed *a* is S or NP.
- longer r. traversal mean more constraints are ignored, more distortion.

Scalability

Flaw: why is accuracy on both types of sentences so low?

- vectors are short
- vectors are only positive
- reconstruction is not done as cleverly as possible
- outer products could be added using Howard-Kahana norming
- ...

Maybe someday this could be broad-coverage, but don't need it today.

Conclusion

This talk defined parsing in terms of (vectorial) associative memory models [Marr, 1971, Anderson et al., 1977, Murdock, 1982, McClelland et al., 1995, Howard and Kahana, 2002]

- one (possibly superposed) activation-based state: cortex as vector
- a set of weight-based cued associations: hippocampus as matrix

Model provides algorithmic-level justification for parallel left-corner parsing.
Model provides algorithmic-level justification for PCFG model.

Model rightly predicts that center embedded sentences are harder to parse.

Model provides an explanatory model of center embedding difficulty:

- due to need to reconstruct active category after constraints on awaited.

Thank you!

Bibliography I

Anderson, J. A., Silverstein, J. W., Ritz, S. A., and Jones, R. S. (1977).
Distinctive features, categorical perception and probability learning: Some applications of a neural model.
Psychological Review, 84:413–451.

Fossum, V. and Levy, R. (2012).
Sequential vs. hierarchical syntactic models of human incremental sentence processing.
In *Proceedings of CMCL 2012*. Association for Computational Linguistics.

Howard, M. W. and Kahana, M. J. (2002).
A distributed representation of temporal context.
Journal of Mathematical Psychology, 45:269–299.

Bibliography II

Marr, D. (1971).
Simple memory: A theory for archicortex.
Philosophical Transactions of the Royal Society (London) B, 262:23–81.

McClelland, J. L., McNaughton, B. L., and O'Reilly, R. C. (1995).
Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory.
Psychological Review, 102:419–457.

Murdock, B. (1982).
A theory for the storage and retrieval of item and associative information.
Psychological Review, 89:609–626.

Bibliography III

Schuler, W., AbdelRahman, S., Miller, T., and Schwartz, L. (2010).
Broad-coverage incremental parsing using human-like memory constraints.
Computational Linguistics, 36(1):1–30.

van Schijndel, M., Exley, A., and Schuler, W. (2013a).
A model of language processing as hierarchic sequential prediction.
Topics in Cognitive Science, 5(3):522–540.

van Schijndel, M., Nguyen, L., and Schuler, W. (2013b).
An analysis of memory-based processing costs using incremental deep syntactic dependency parsing.
In *Proceedings of CMCL 2013*. Association for Computational Linguistics.

Bibliography IV

van Schijndel, M., Schuler, W., and Culicover, P. W. (2014).
Frequency effects in the processing of unbounded dependencies.
In *Proc. of CogSci 2014*. Cognitive Science Society.