TRACX2:
Improving the Plausibility of TRACX, a Connectionist Framework for Sequence Segmentation and Chunk Extraction

Or

“Nice model, Bob, but what’s an IF-THEN-ELSE statement doing in a connectionist model, huh?”

Robert M. French
(CNRS-LEAD)
Denis Mareschal
(Birkbeck University of London)

The problem of segmenting continuous speech

Infants can segment speech, basically without semantics, by 8 months.

How?

A bit of background on Transitional Probabilities, the Standard View of how segmentation is done

Within-word vs. Between-word Transitional Probabilities

Suppose a language contains 5 words: *abc, def, ghi, jkl, mno*

***a* is always followed by *b:* P(b|a) = 1 ,**

Within-word vs. Between-word transitional probabilities

Suppose the language is made up of 5 words: *abc, def, ghi, jkl, mno*

a b c g h i m n o j k l a b c g h i d e f a b

***a* is always followed by *b:* P(b|a) = 1 ,**
***b* is always followed by c: P(c|b) = 1**

Within-word vs. Between-word transitional probabilities

Suppose the language is made up of 5 words: *abc, def, ghi, jkl, mno*

a b c g h i m n o j k l a b c g h i d e f a b

***a* is always followed by *b:* P(b|a) = 1**
***b* is always followed by c: P(c|b) = 1**

But *c* is only followed by g: P(g|c) = 0.2

Strong claim by Saffran et al., Aslin et al., etc.

Sensitivity to differences between within-word and between-word TPs is sufficient for sequence segmentation and chunk extraction.

Words versus Partwords

Words are (frequently) defined as 2- or 3-syllable groups where the internal TPs are 1. Words in our sequence are: *abc def ghi jkl mno*

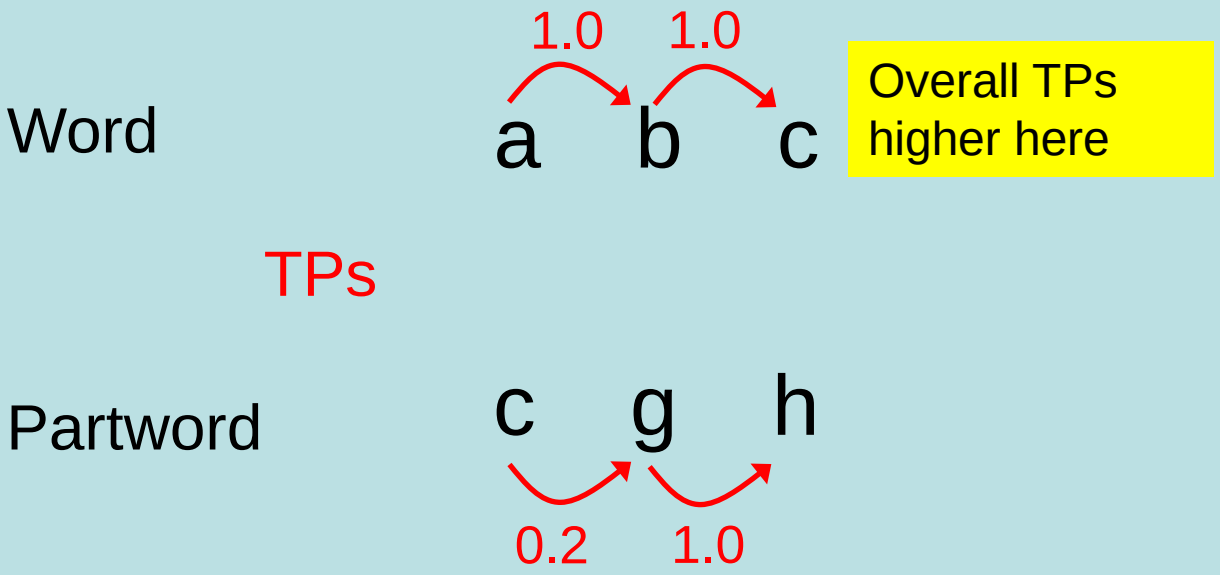
a b c g h i m n o j k l a b c g h i d e f a b c j k

Partwords are defined as 2- or 3-syllable groups whose first syllable is the final syllable of one word and the leading syllable(s) of the following word.

a b c g h i m n o j k l a b c g h i d e f a b c j k

cgh imn

Higher TPs → Greater association → Better learning



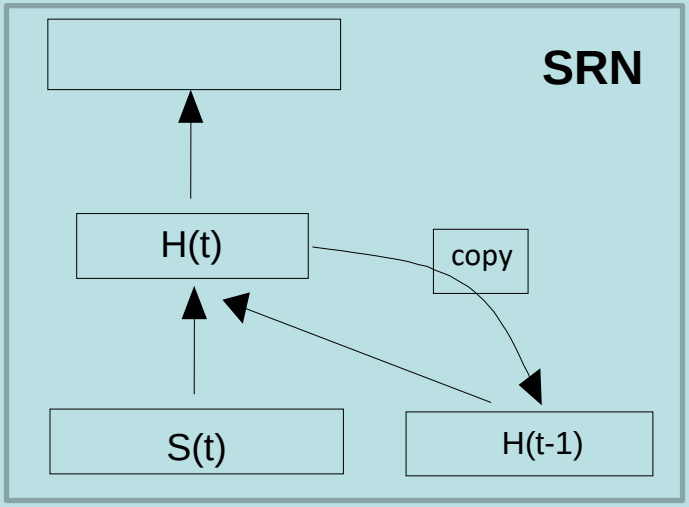
Conclusion: abc will be learned better than cgh.

The Simple Recurrent Network (SRN; Elman, 1990) model of word segmentation

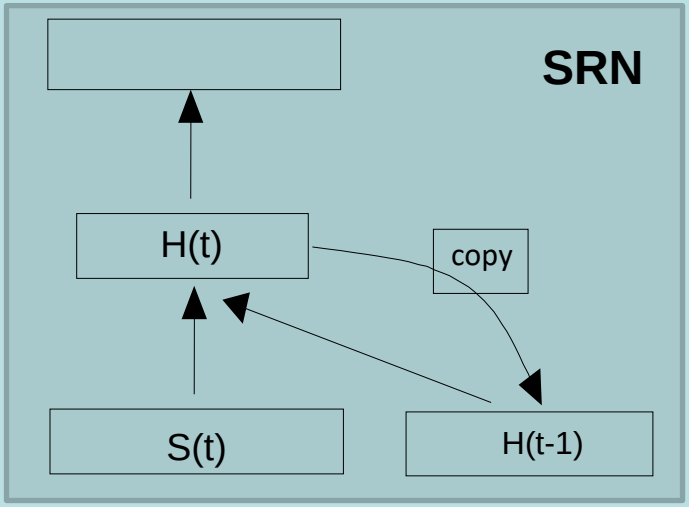
Principles of the SRN model

- The higher the TP between two syllables, the better the prediction of the upcoming syllable based on the present syllable.
- SRNs learn by bringing into alignment predictions and what actually occurs.
- Word boundaries are where prediction is poorest.

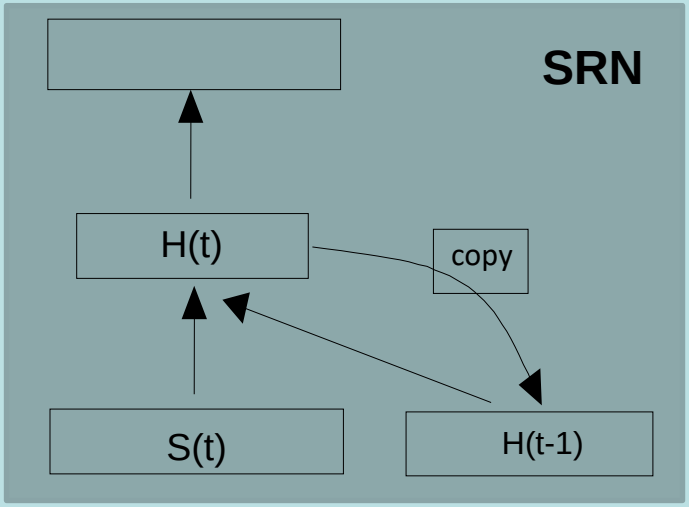
Sequence of items: S(1), S(2), S(3), … S(t), S(t+1), …, S(n)



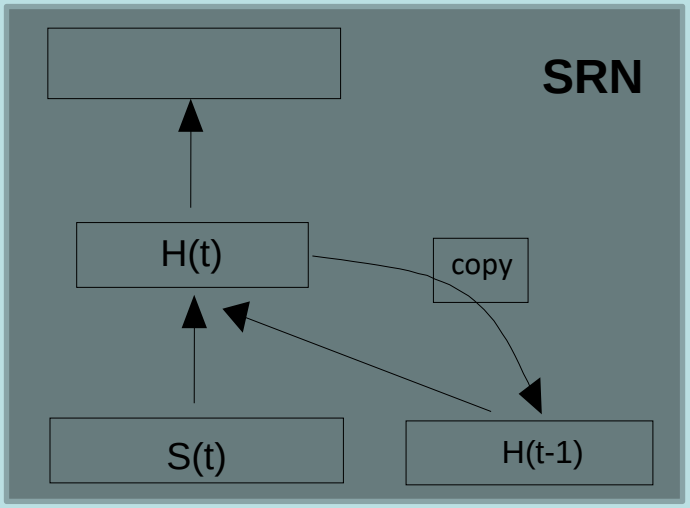
Sequence of items: S(1), S(2), S(3), … S(t), S(t+1), …, S(n)



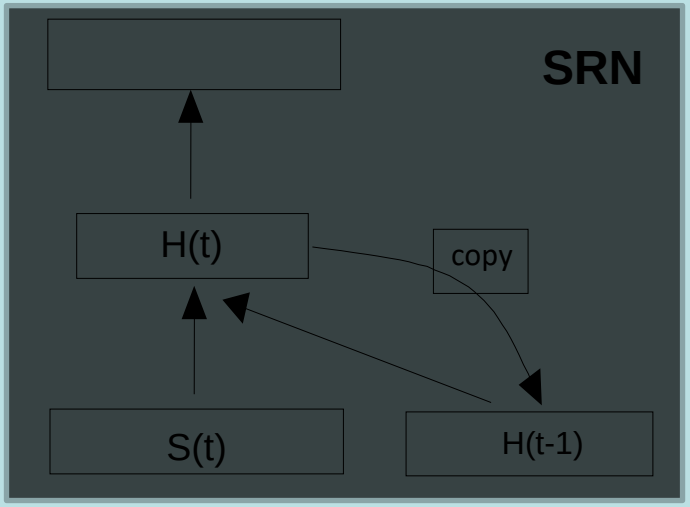
Sequence of items: S(1), S(2), S(3), … S(t), S(t+1), …, S(n)



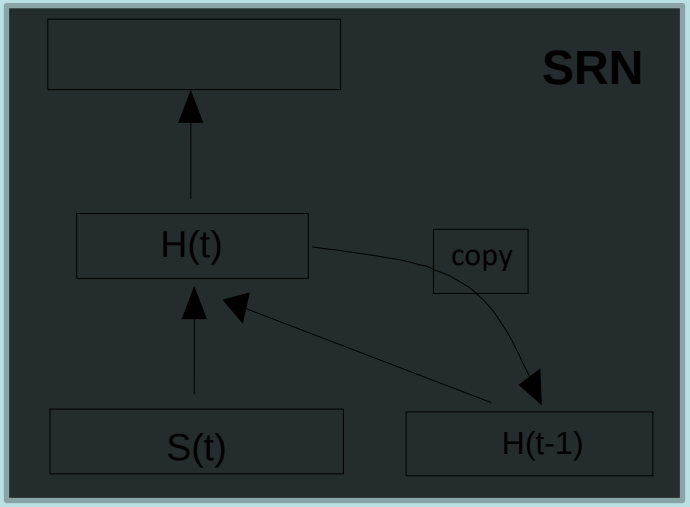
Sequence of items: S(1), S(2), S(3), … S(t), S(t+1), …, S(n)



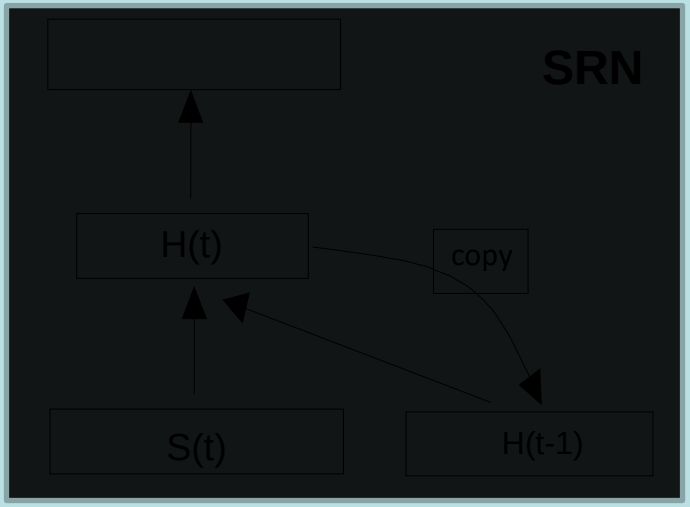
Sequence of items: S(1), S(2), S(3), … S(t), S(t+1), …, S(n)



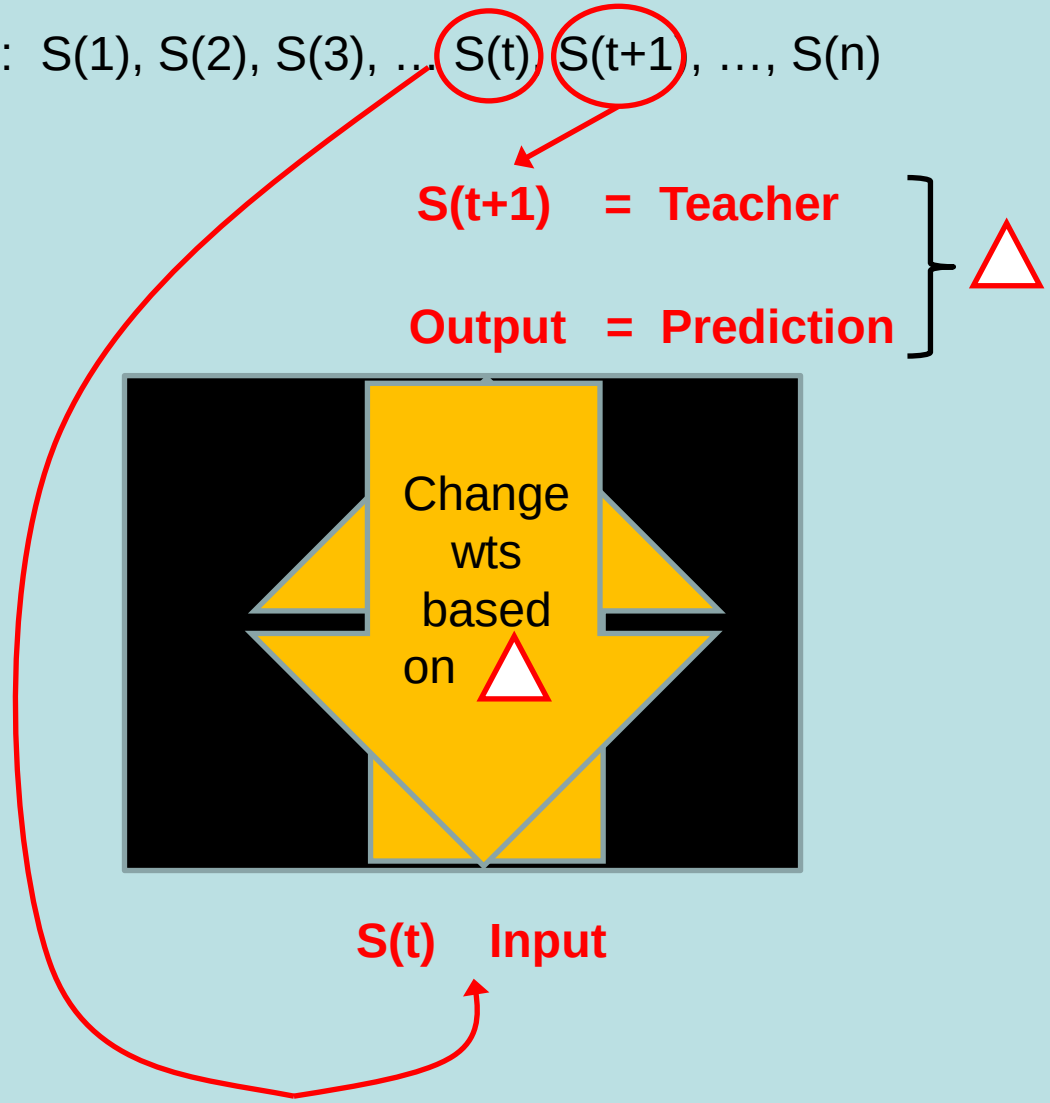
Sequence of items: S(1), S(2), S(3), … S(t), S(t+1), …, S(n)



Sequence of items: S(1), S(2), S(3), … S(t), S(t+1), …, S(n)



Sequence of items: S(1), S(2), S(3), … S(t), S(t+1), …, S(n)



For *a b c g h i m n o j k l a b c g h i …* the SRN will quickly learn that:

a predicts *b* and *b* predicts *c* (*abc*)
g predicts *h* and *h* predicts *i* (*ghi*)
etc.

The SRN, therefore, learns Words *abc def ghi, …*
much better than Partwords *cgh imn ojk, …*

A fly in the ointment:
Backward Transitional Probabilities

A backward TP is the probability that, for a given syllable B, it is *preceded* by A. Backward TPs are also cues for segmentation.

Forward and Backward TPs

- Forward TP: Given a *q*, the probability that it is *followed* by a *u* is 1 (very predictive)
- Backward TP: But, given a *u*, the probability that it is *preceded* by a *q*, is only 0.01 (not predictive).

Consider *ez* in French (“Parl*ez*-vous français?”)

Forward TP: Given an *e,* the probability that it is followed by a *z* is 0.03. (not predictive)
Backward TP: Given a *z*, the probability that it is preceded by an *e* is 0.84. (very predictive) The Backward TP is a far better cue than the Forward TP.

Backward transitional probabilities

Perruchet built a vocabulary based ONLY on backward TPs.

He discovered that adults segment words well above chance with only backward TPs.

Pelucchi et al. (2009) and French et al. (2011) confirmed this result in infants.

So what?

The problem is that an SRN relies on predicting the NEXT item in a sequence (i.e., FORWARD TPs) in order to learn.

When an SRN was tested on Perruchet's backward TP data, it failed (French et al., 2011). It learned partwords better than words.

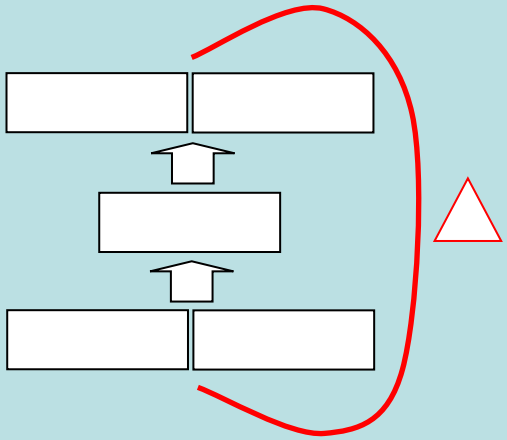
In addition, how does an SRN "know" that a particular TP is lower than previous ones, thereby signaling a word boundary? Where is the TP information stored and compared with other TPs?

Enter **TRACX**:
A connectionist, memory-based way of looking at segmentation

TRACX

- TRACX is a connectionist recursive autoencoder memory model of sequence segmentation and chunk extraction.
- It does not rely on prediction, as the SRN does.
- It recognizes chunks of syllables it has previously encountered.
- It dynamically re-uses the chunks of syllables that it has discovered.
- It has no Working Memory in which chunks that have been found are explicitly stored and manipulated (cf. **PARSER**, Perruchet & Vinter, 1998).
- Forgetting and interference emerge from the architecture and are not hand-coded, as they are in **PARSER**.
- It generalizes well to new input.

Autoencoders recognize what they have seen before

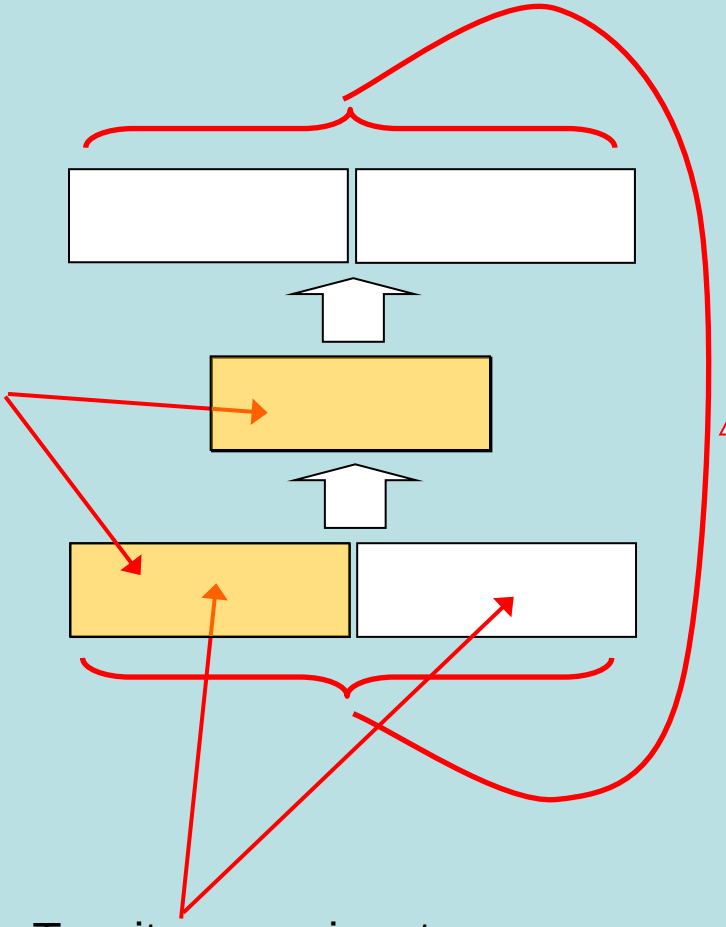


Autoassociators are designed to answer this question:
"Have I encountered this input before?"

△ large = no

△ small = yes

TRACX

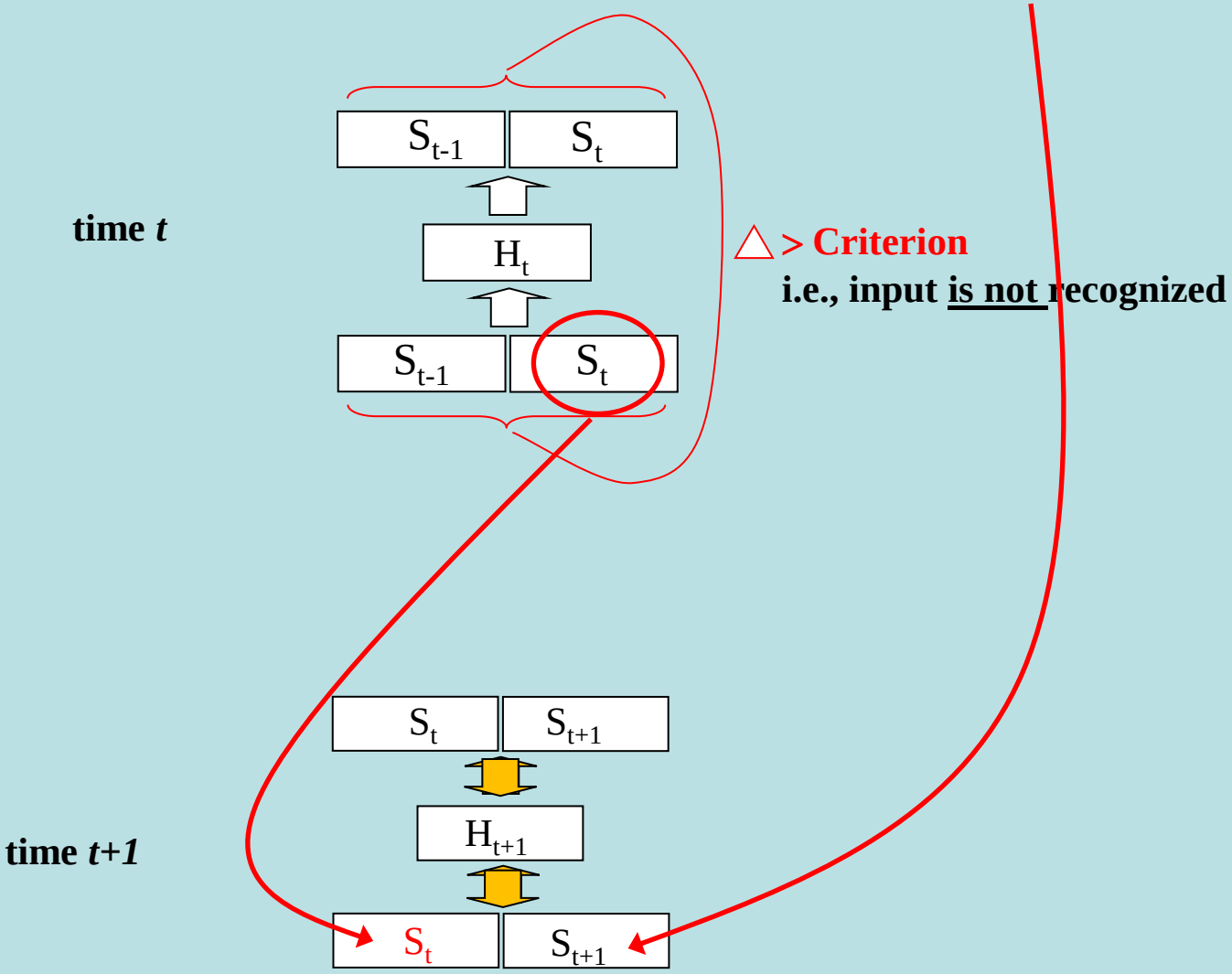


Hidden layer is exactly half the size of the input layer

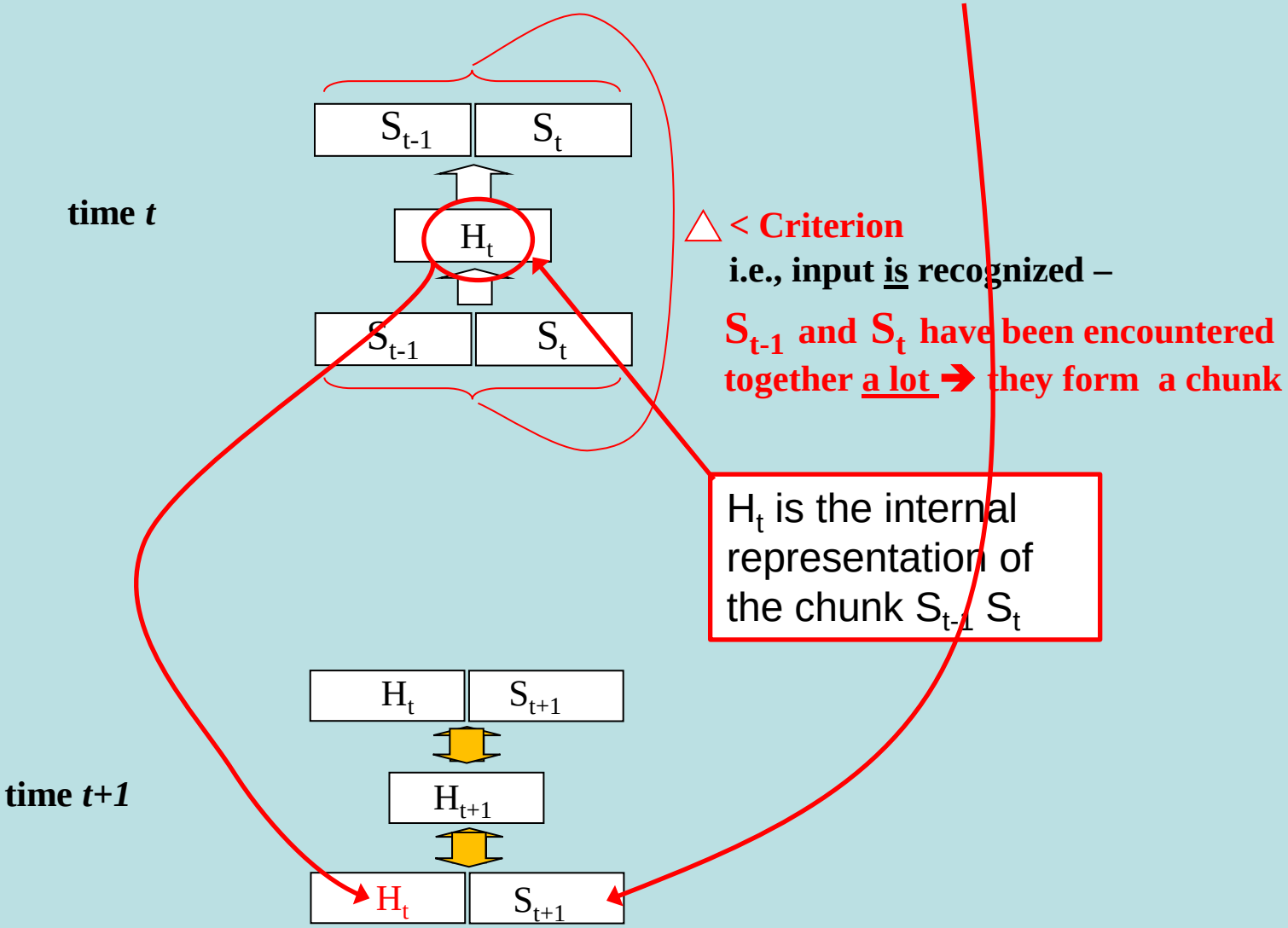
It compares its input to its output

Two items on input

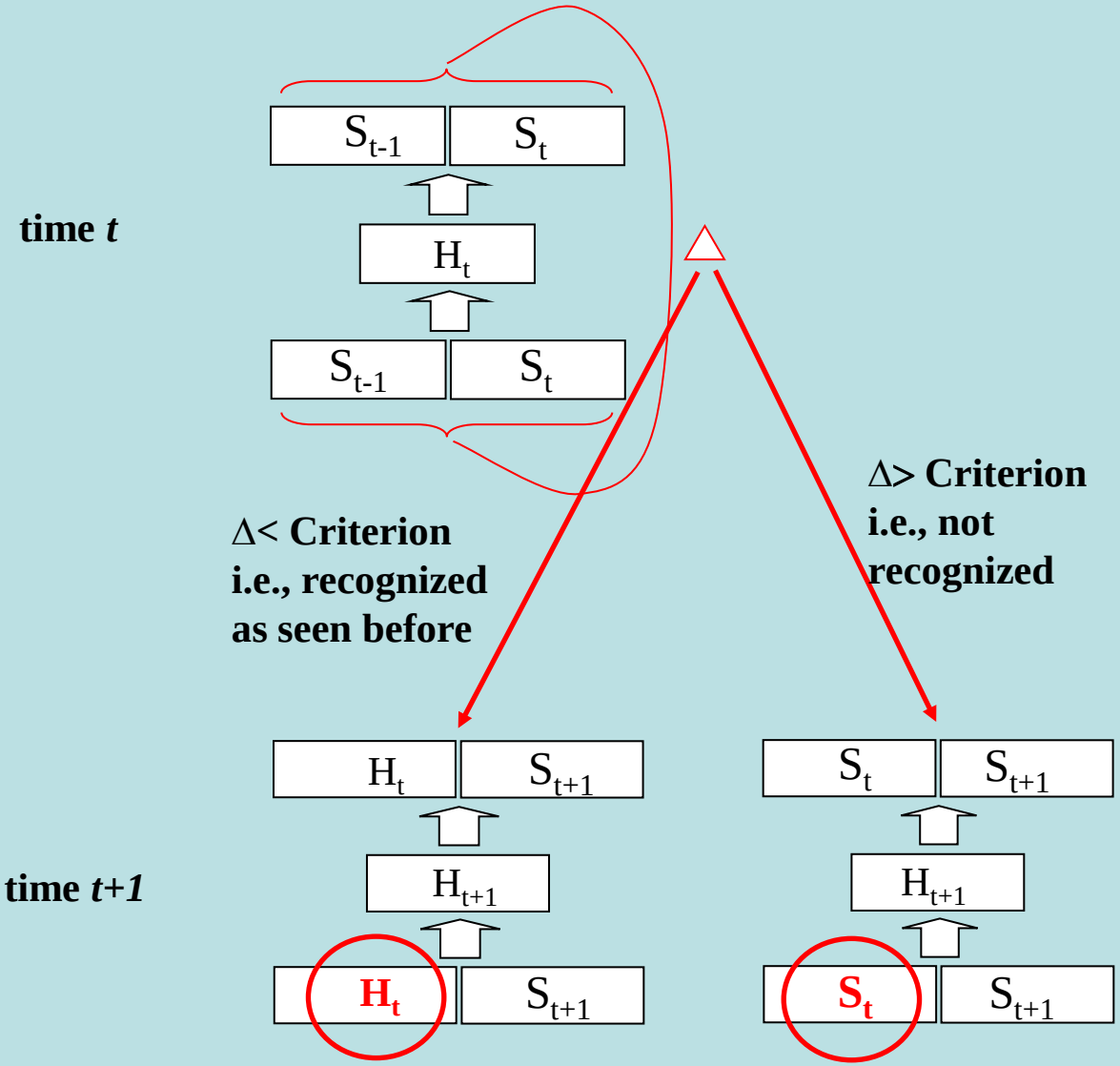
Consider the syllable sequence: S_1 S_2 S_3 …. S_t S_{t+1} . . .



Consider the syllable sequence: S_1 S_2 S_3 …. S_t S_{t+1} . . .



Consider the syllable sequence: S_1 S_2 S_3 …. S_t S_{t+1} . . .



Results

TRACX reproduces a wide range of empirical data

- Saffran et al. (1996)
- Aslin et al. (1998)
- Frank et al. (2010), two experiments
- Perruchet & Desaulty (2008), two experiments
- Giroux & Rey (2009)
- Equal TP

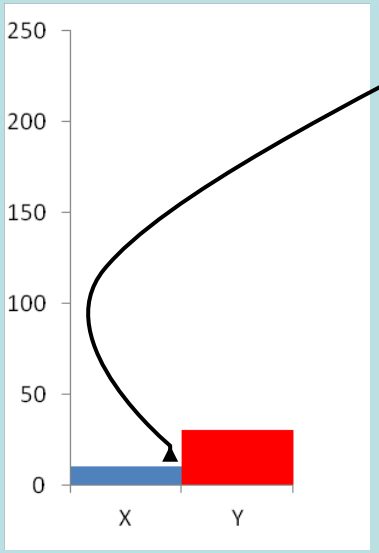
Scaling up:

- Brent & Cartwright (1996) infant-directed language corpus

Generalization

- Bilingual micro-language learning (French, 1998)

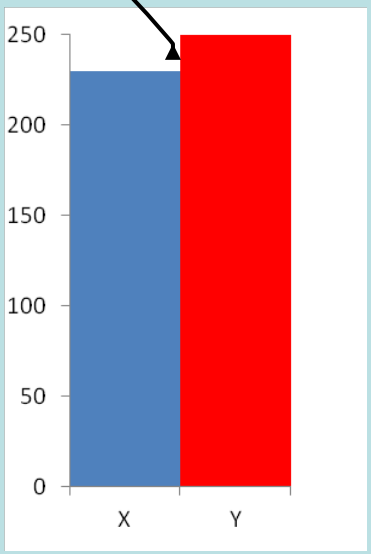
Proportional Difference:



The absolute difference between X and Y is the same in both graphs,

but

the <u>relative</u> difference between X and Y is much greater on the left.



$$proportional\ difference = \frac{Y - X}{Y + X}$$

Proportional Difference allows us to compare Looking Times (babies) to Output Error (TRACX).

$$proportional\ difference(babies) = \frac{LookingTime(Partwords) - LookingTime(Words)}{LookingTime(Partwords) + LookingTime(Words)}$$

$$proportional\ difference(TRACX\ 2) = \frac{OutputError(Partwords) - OutputError(Words)}{OutputError(Partwords) + OutputError(Words)}$$

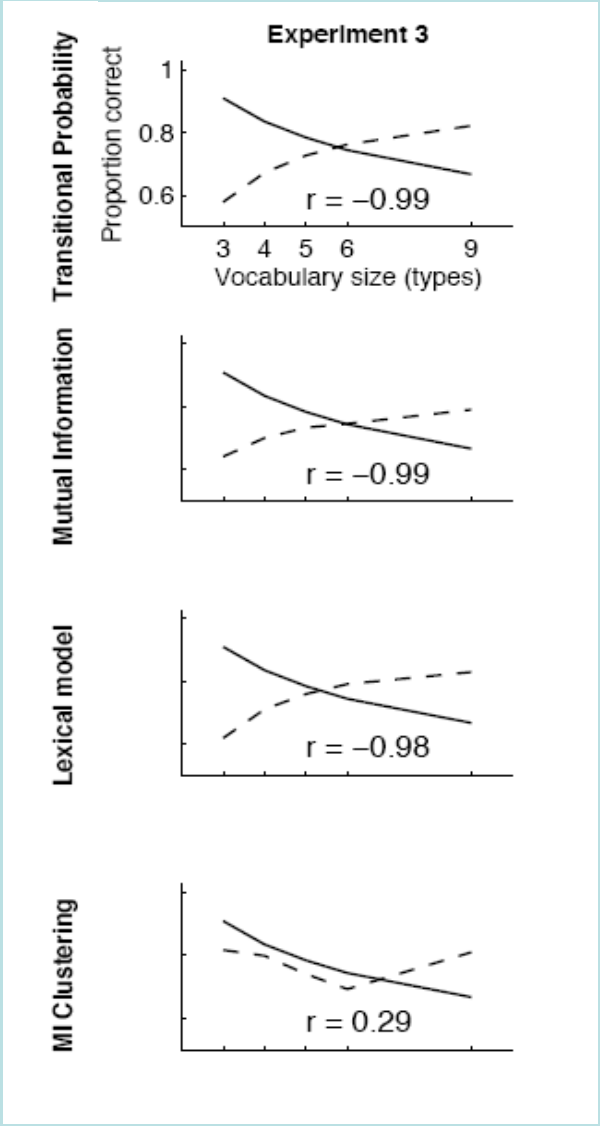
Comparison of Humans, TRACX and an SRN across 5 based on proportional differences.

	Words learned significantly better than Partwords?		
	Humans	TRACX	SRN
Saffran et al. (1996)	Yes	Yes	Yes
Aslin et al. (1998)	Yes	Yes	Yes
Perruchet & Desaulty (2008). Expt. 1	Yes	Yes	Yes
Perruchet & Desaulty (2008). Expt. 2	Yes	Yes	**No**
Equal TP (French et al. (2011)	Yes	Yes	Only Slightly

As overall vocabulary size gets bigger, word segmentation gets harder. (Frank et al., 2010)

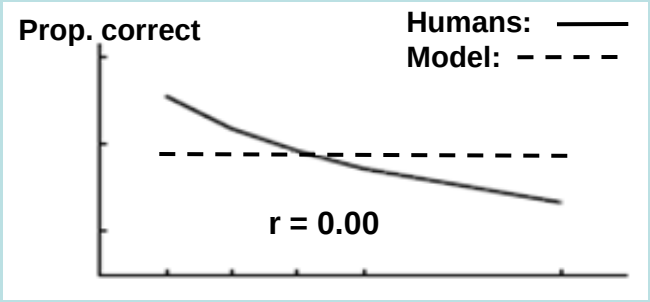
Human data ——————

Model data - - - - -

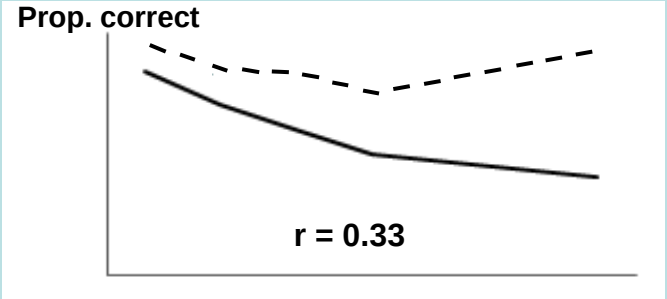


Effect of Vocabulary Size

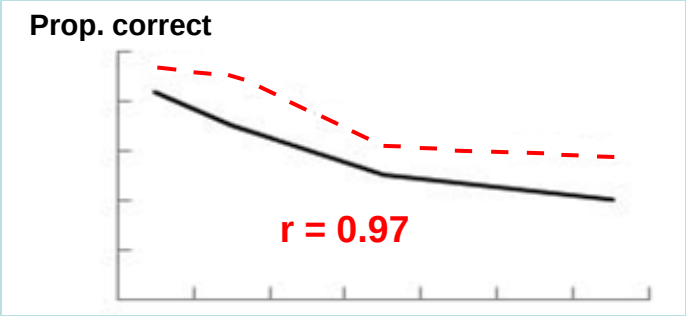
PARSER



SRN



TRACX



Generalization and discovering structure in TRACX

Two micro-languages, Alpha and Beta

- Alpha and Beta each consist of three-syllable words.
- All words have one Initial, Middle, and Final syllable.
- Alpha syllables: Initial: {a, b, c}, Middle: {d, e, f}, and Final: {g, h, i}.
- Beta syllables: Initial: {d, e, f}, Middle: {a, b, c}, and Final: {g, h, i}.
- Language switching probability p = 0.025.
- No markers indicating either word boundaries or language boundaries.

A typical language training sequence of syllables looked like this:

aegcfgcdhbdhbfgbfgbdgafifagechfaiecgecifaifbheagdaif ...

Alpha

Beta

These Alpha words were left out of the training sequence

b e f

c f i

a d g

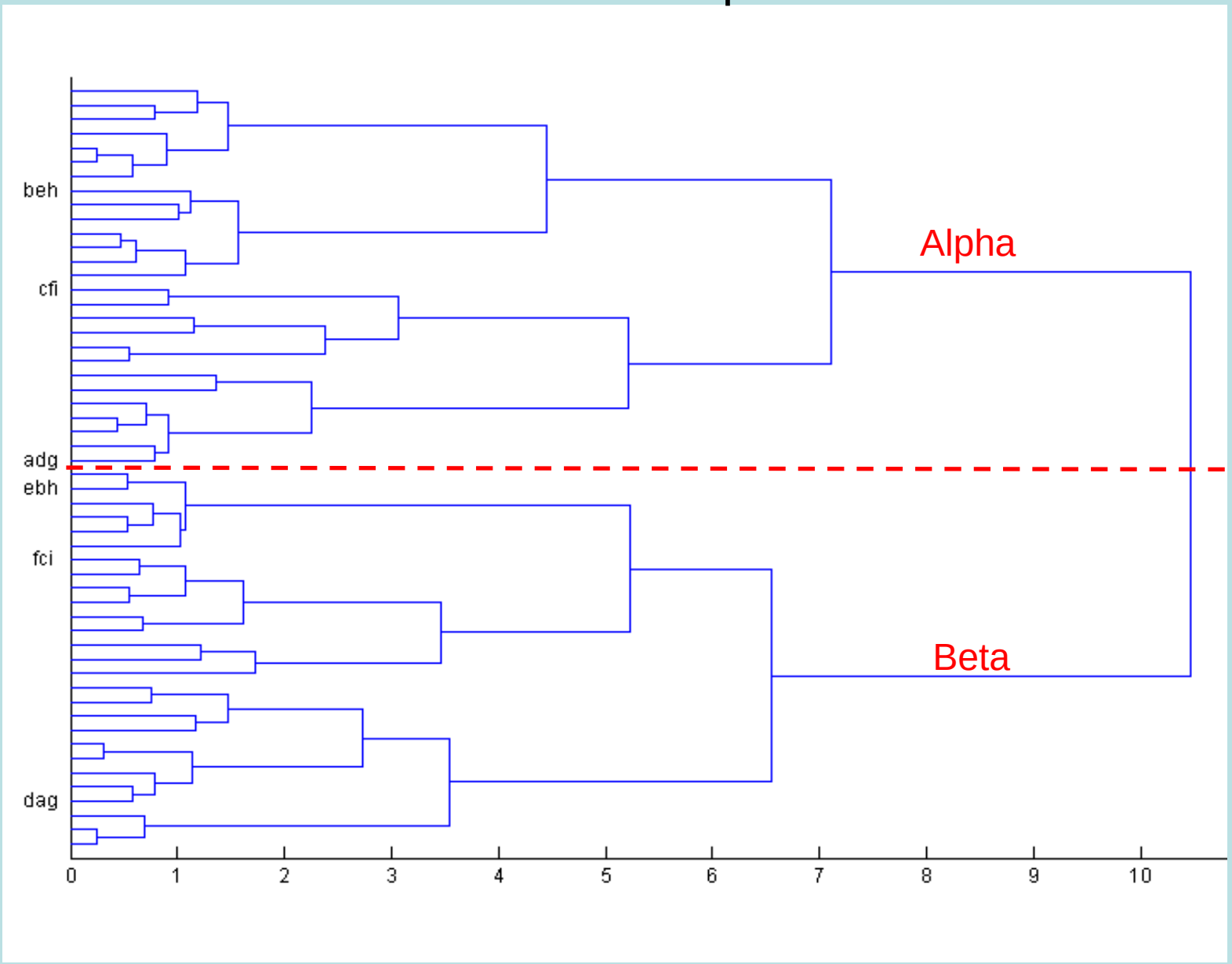
These Beta words were left out of the training sequence

e b f

f c i

d a g

TRACX's internal representations

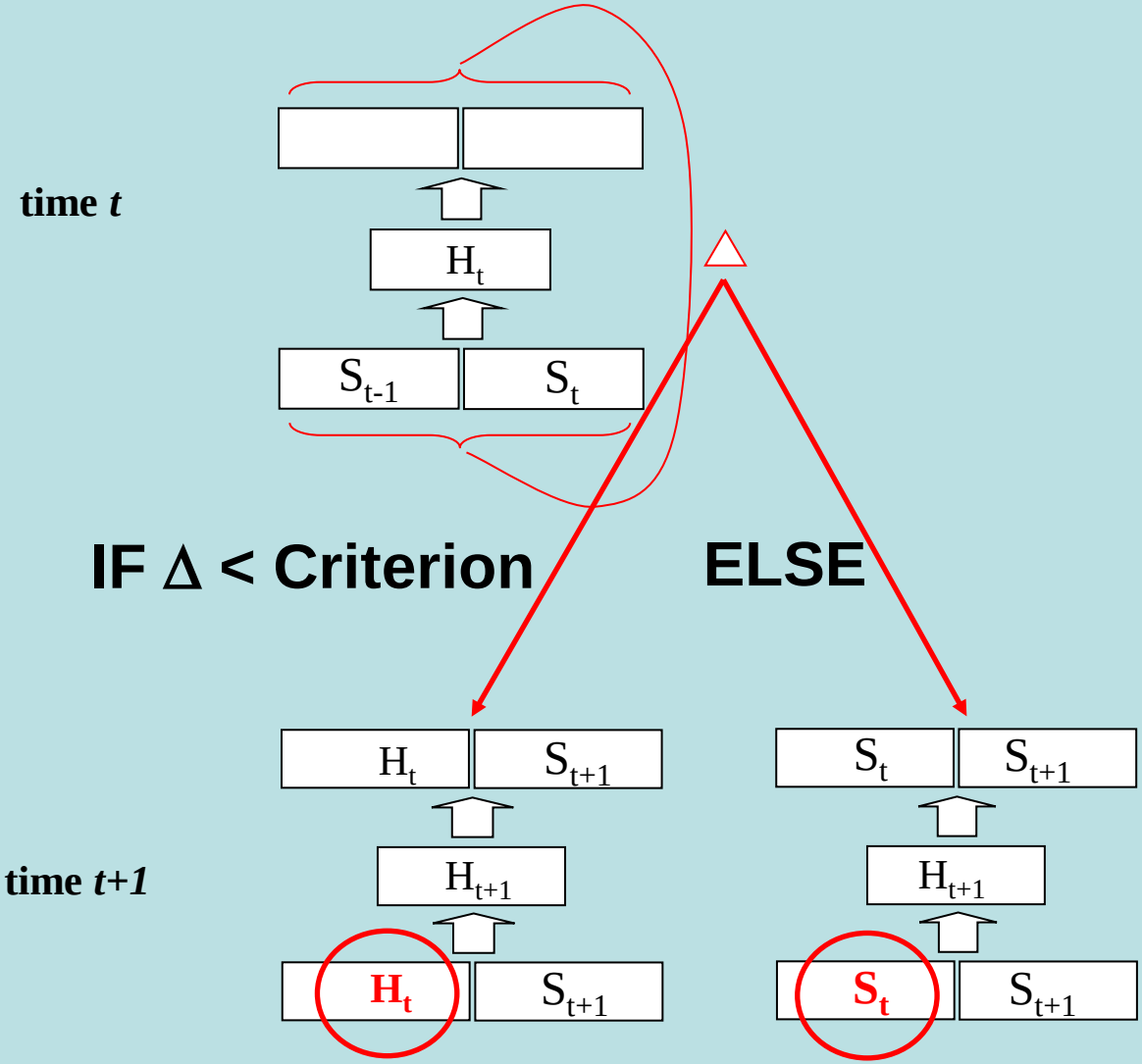


Enter GARY COTTRELL…

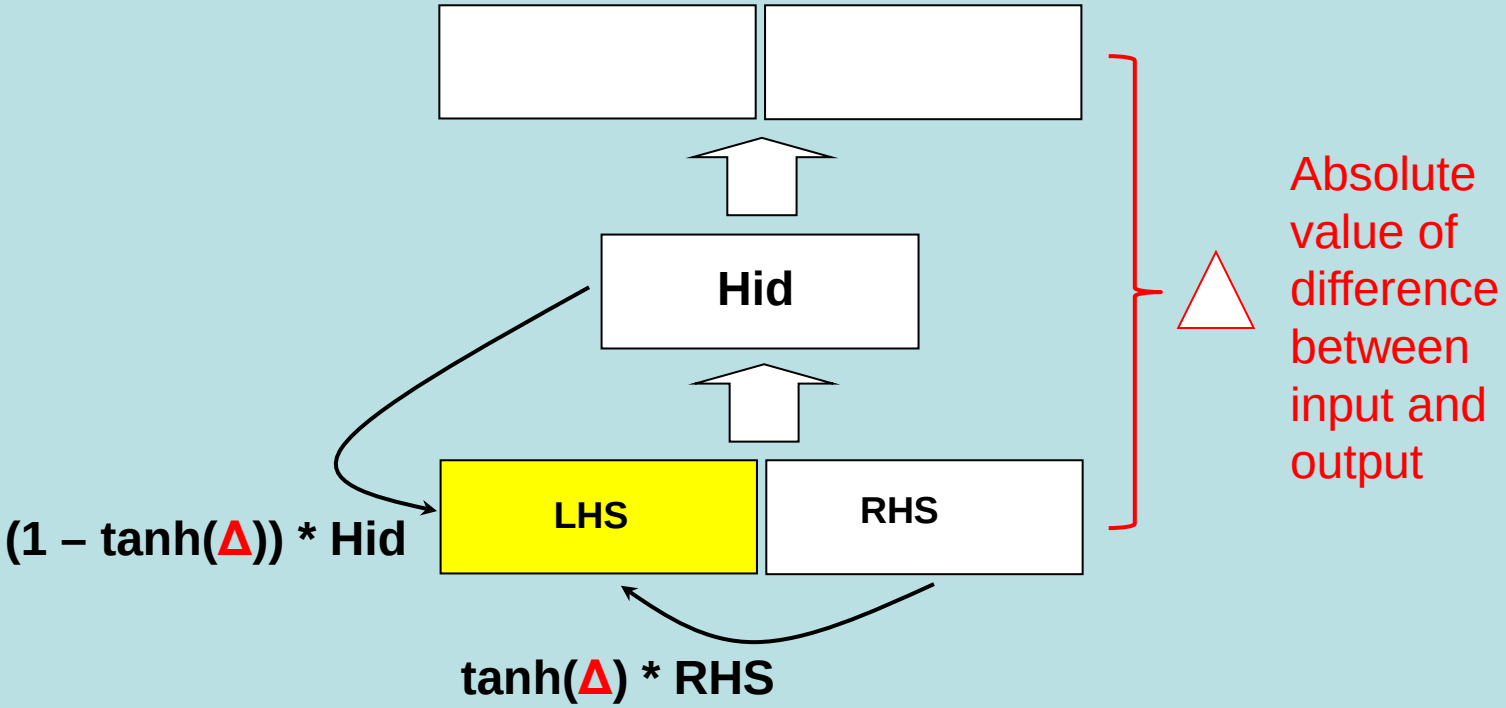
… The Last Hippie

“Nice model, Bob, but what’s an IF-THEN-ELSE statement doing in a connectionist model, huh?”

For the syllable sequence: S_1 S_2 S_3 …. S_{t-1} S_t S_{t+1} . . .

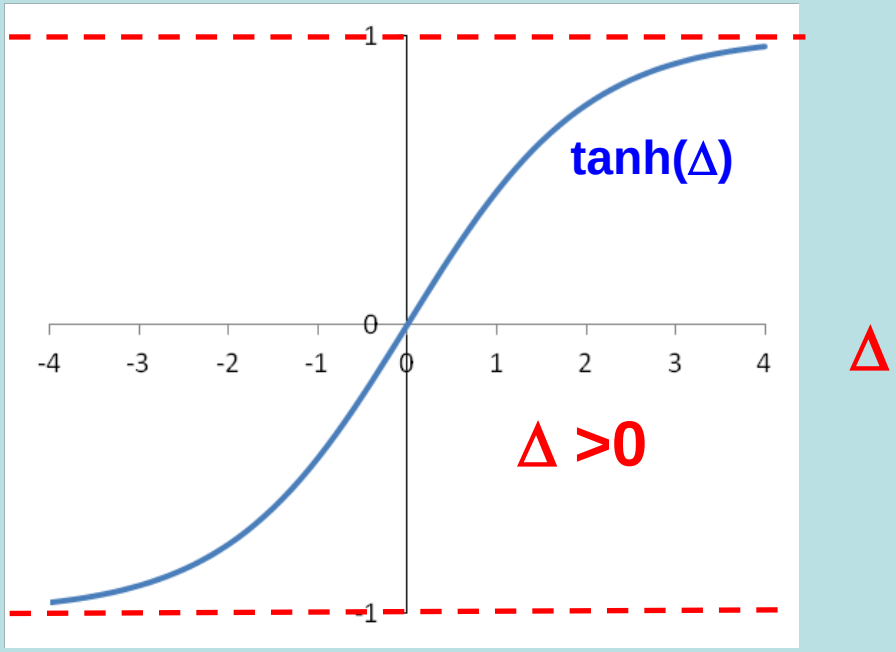


The solution: TRACX 2.0



LHS = (1 – tanh(Δ)) * Hid + tanh(Δ) * RHS

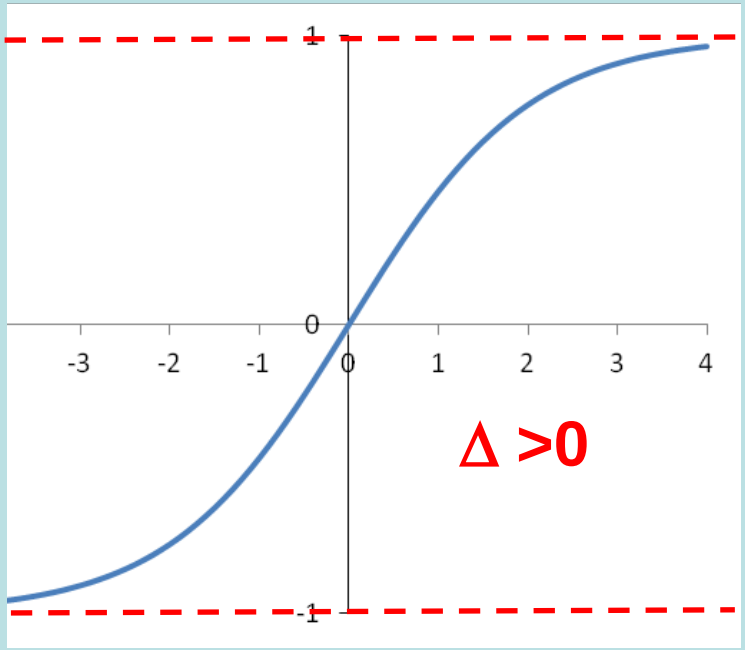
A crash course in tanh



- tanh(x) «squashes» input between -1 and 1

- Δ is always positive, so we can remove the left-hand side of the graph.

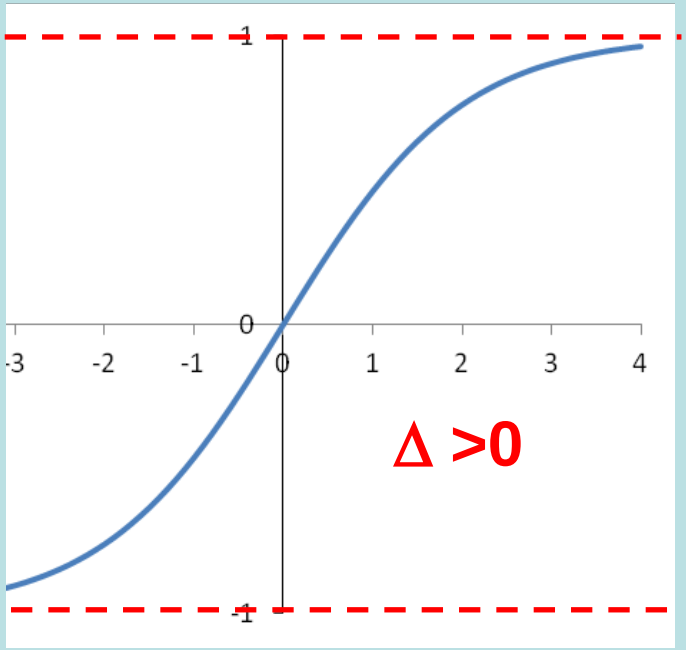
A crash course in tanh



- tanh(x) «squashes» input between -1 and 1

- Δ is always positive, so we can remove the left-hand side of the graph.

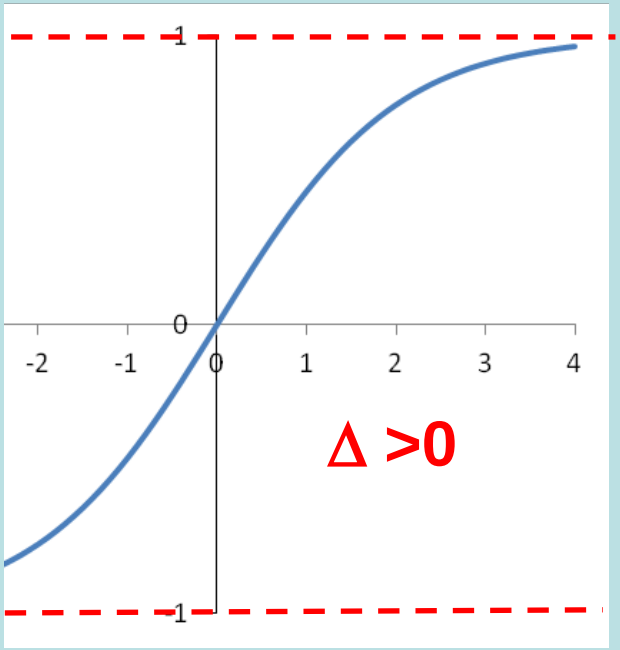
A crash course in tanh



- tanh(x) «squashes» input between -1 and 1

- Δ is always positive, so we can remove the left-hand side of the graph.

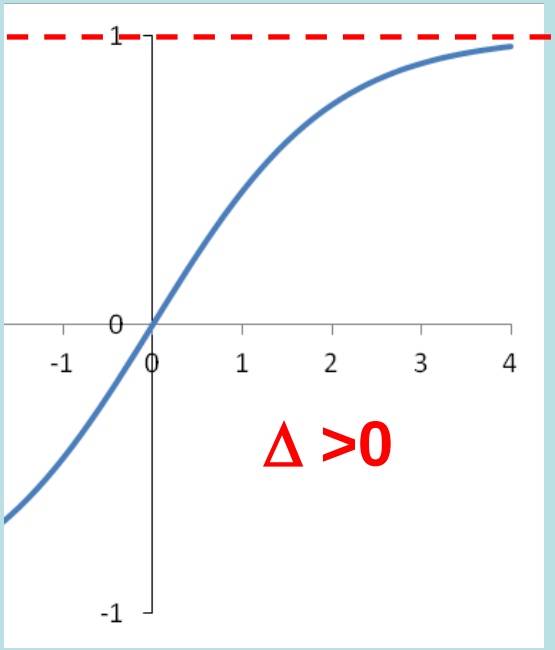
A crash course in tanh



- tanh(x) «squashes» input between -1 and 1

- Δ is always positive, so we can remove the left-hand side of the graph.

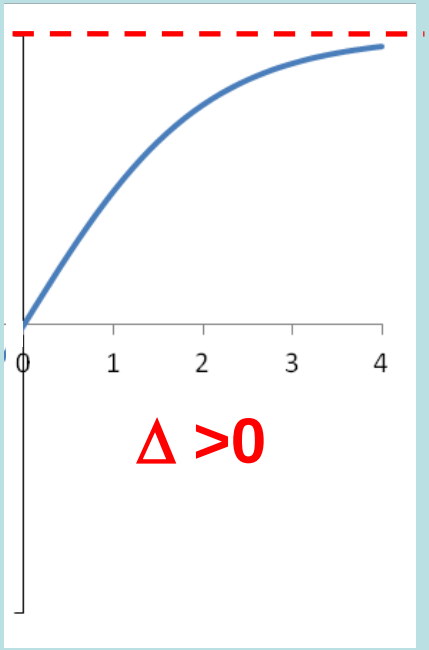
A crash course in tanh



- tanh(x) «squashes» input between -1 and 1

- Δ is always positive, so we can remove the left-hand side of the graph.

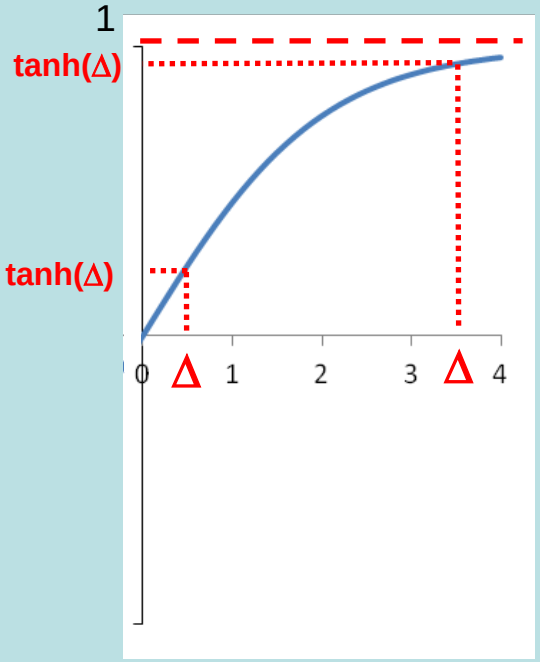
A crash course in tanh



- tanh(x) «squashes» input between -1 and 1

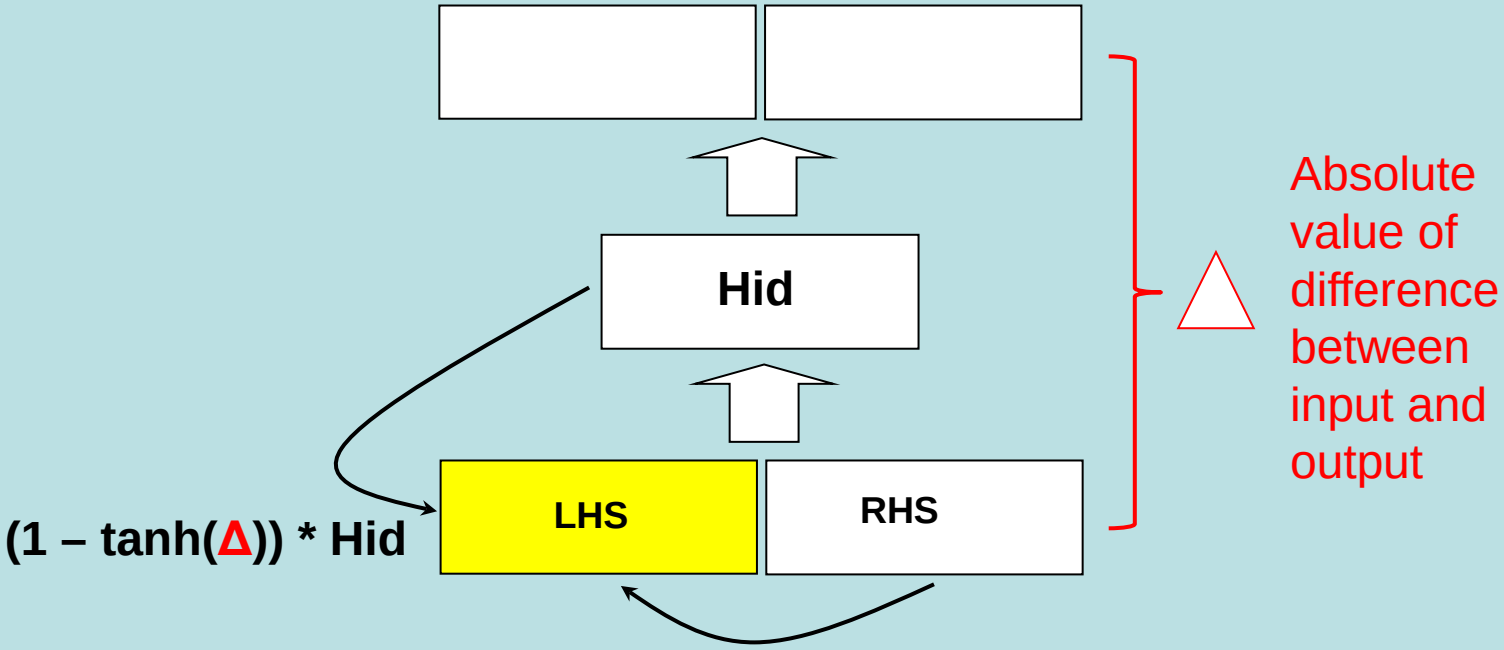
- Δ is always positive, so we can remove the left-hand side of the graph.

A crash course in tanh



Small values of Δ give small values of tanh(Δ)

Large values of Δ give values of tanh(Δ) ≈ 1



LHS = (1 – tanh(Δ)) * Hid + tanh(Δ) * RHS

Δ large ("I don't recall seeing these items together"), tanh(Δ) is ≈ 1. Most of the contribution to LHS is from RHS.

Δ small ("These two items have been together a lot. They must be a chunk.") tanh(Δ) is ≈ 0. Most of the contribution to LHS is from Hidden Layer

LHS = (1 – tanh(Δ)) * Hid + tanh(Δ) * RHS

solves the IF-THEN-ELSE problem

But it also solves the problem of the **graded learning** of chunks.

In TRACX (and in most of AI) chunks are All-or-Nothing entities.

But that is wrong. Chunks become stronger with exposure. The above technique implements this gradual increase in chunk strength.

Poorly chunked items (you still hear the component words):

- Smartphone
- Carwash
- Petshop

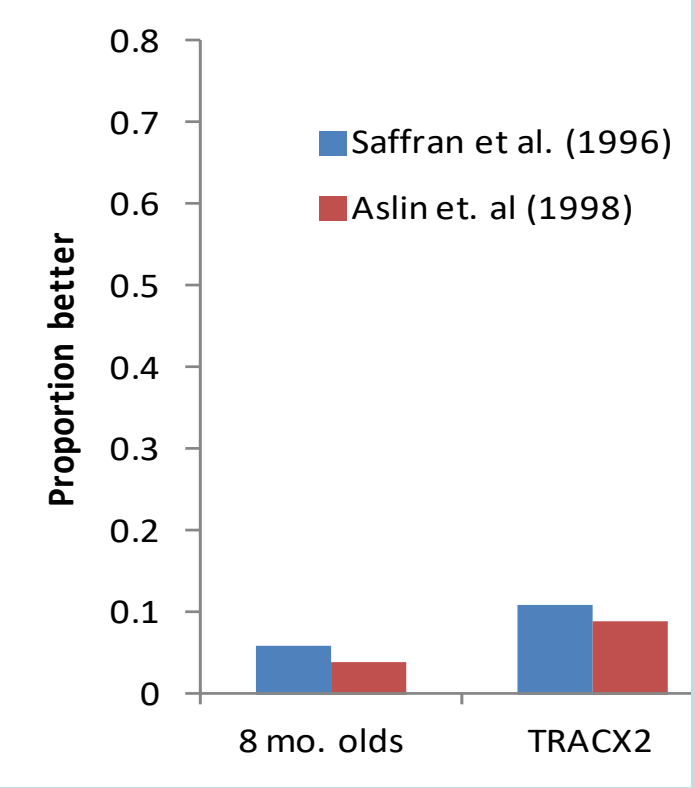
Moderately chunked items:

- Sunburn
- Heartbeat
- Overhang (for climbers!)

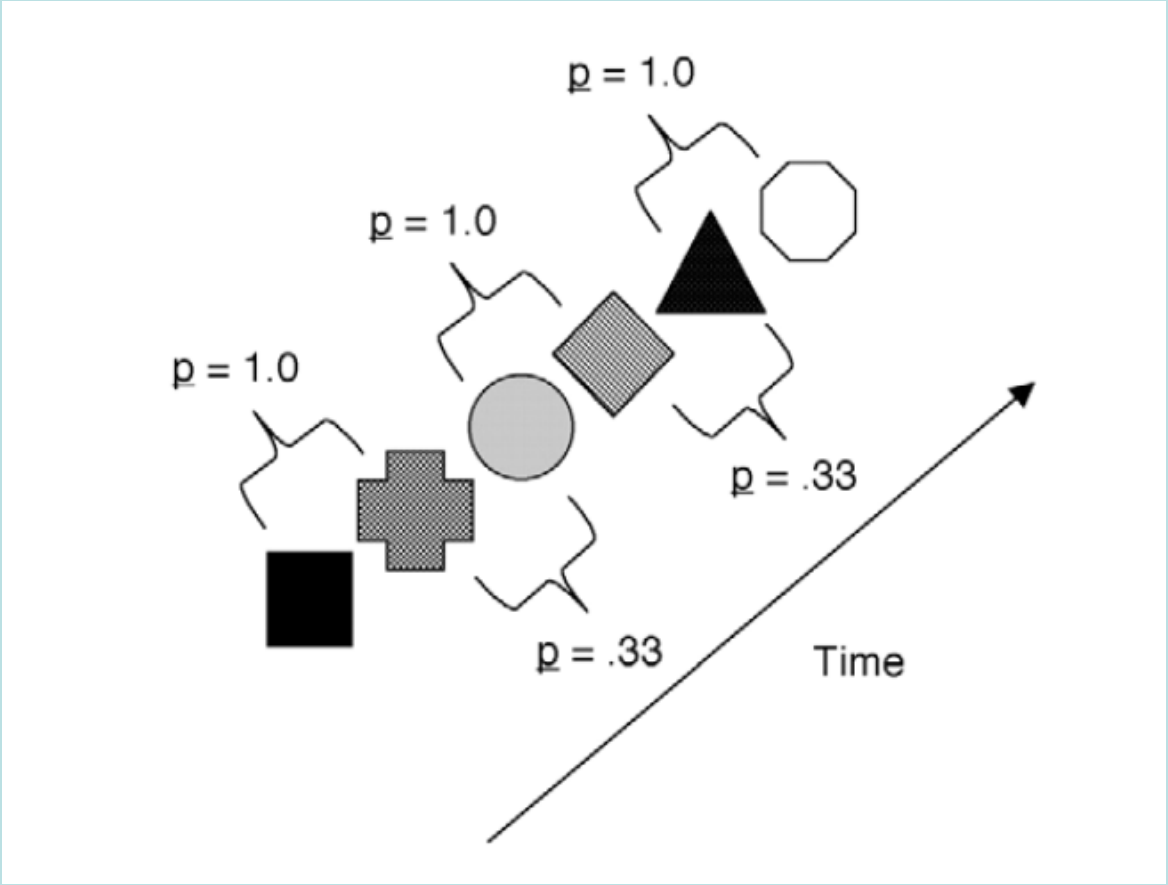
Completely chunked items:

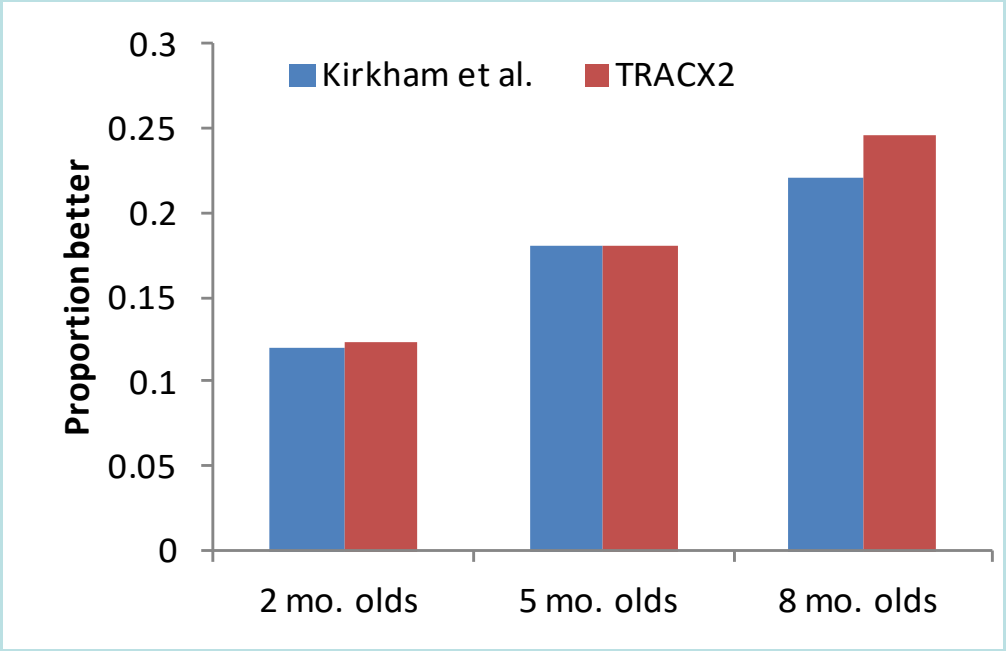
- Football
- Cupboard
- Correlate
- Automobile

Syllable Sequences



Images, not syllables
Kirkham et al., 2002. 6 image items, with TPs as shown.





Learning rates:

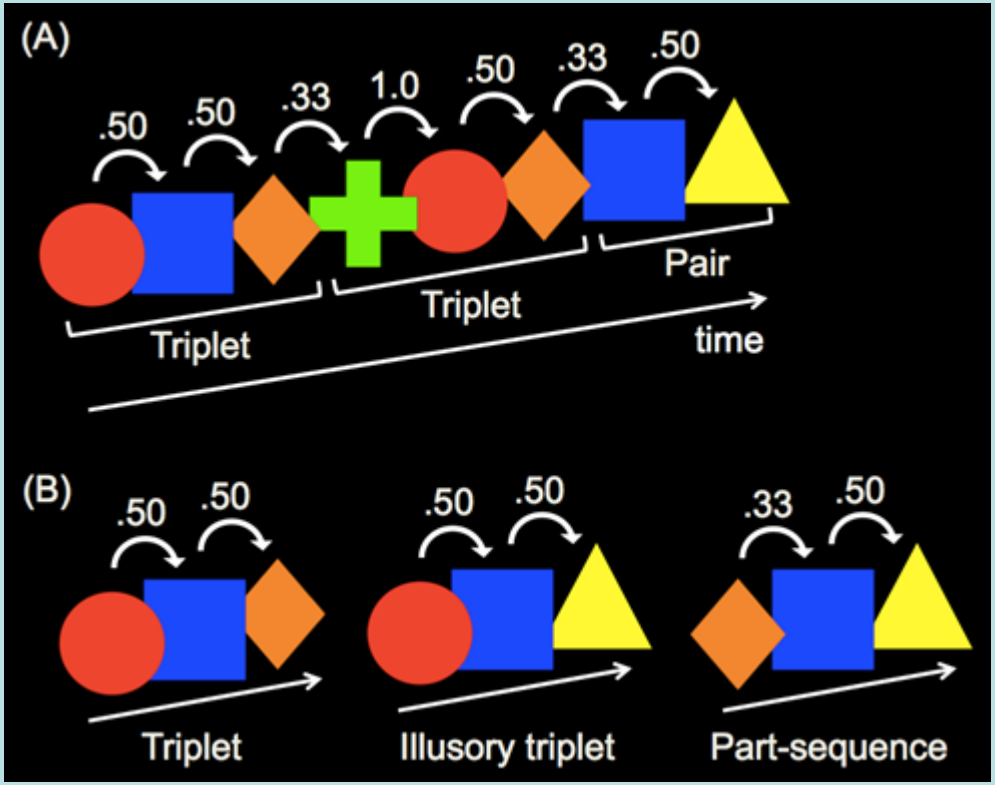
2 mo. olds: 0.0005

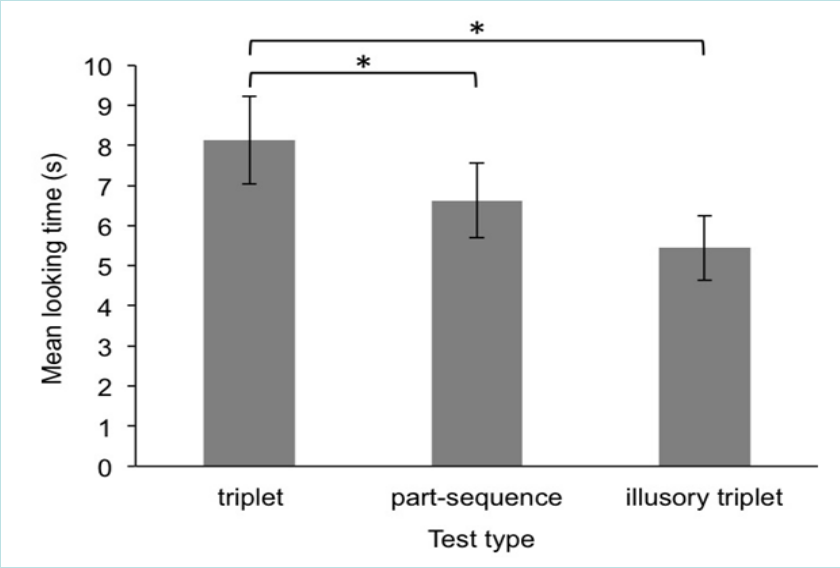
5 mo. olds: 0.0015

8 mo. olds: 0.005

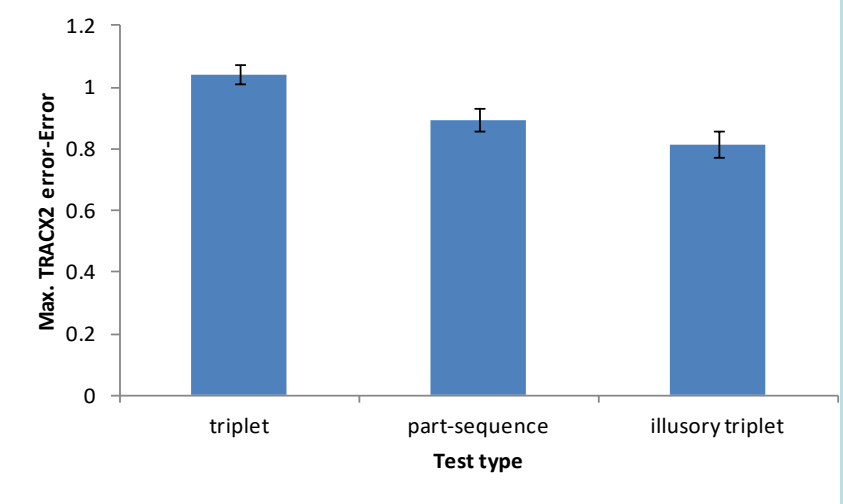
Embedded sequences

Slone & Johnson (2016)





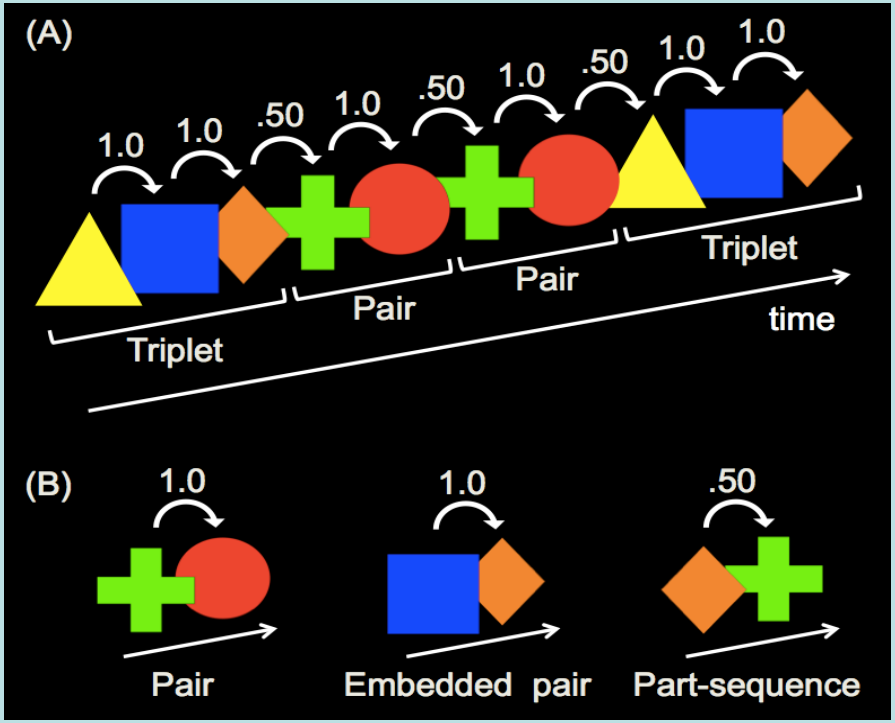
Slone & Johnson (2016)

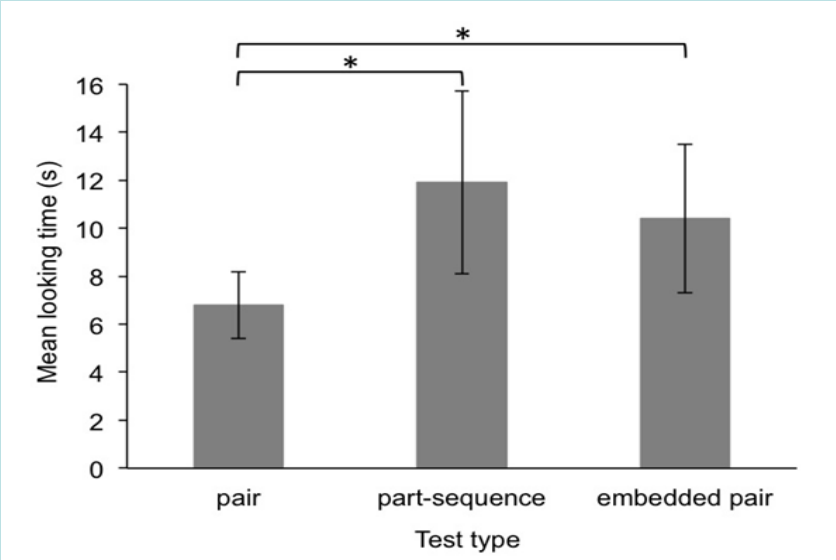


TRACX2

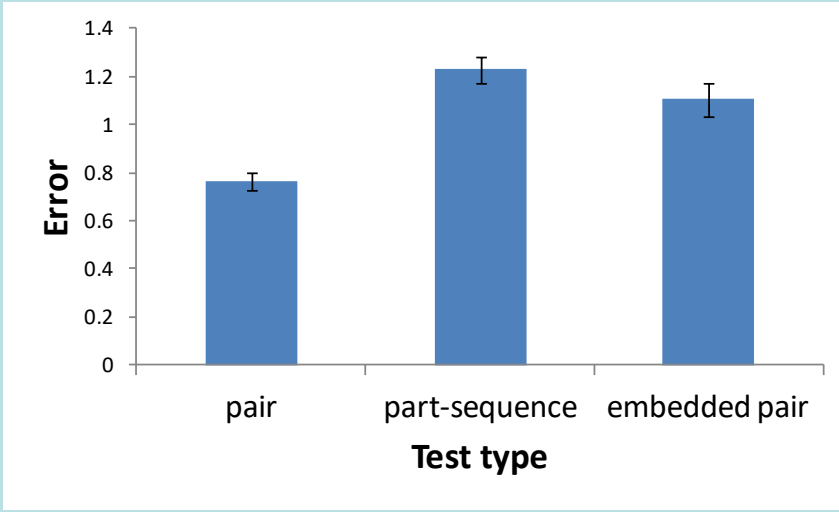
Embedded sequences

Slone & Johnson (2016)





Slone & Johnson (2016)



TRACX2

Conclusions

- TRACX and TRACX2 are very simple recursive encoder models, based on the recognition of previously encountered items and not on the prediction of upcoming items.
- They account for a wide range of empirical data in sequence segmentation in both adults and infants.
- They do not require chunks to be explicitly stored in Working Memory or numerous full scans of WM, as required by PARSER (Perruchet & Vinter, 1998).
- They handle backward transitional probability cues, which an SRN cannot.
- They can extract elementary structure from the sequences they are analyzing and may be able to extract more complex (e.g., grammatical) structures from their input.

This talk in pdf format can be found here:

http://leadserv.u-bourgogne.fr/fr/membres/robert-m-french

Tab: “En savoir plus”

The TRACX and TRACX2 papers can be found here:

http://leadserv.u-bourgogne.fr/fr/membres/robert-m-french

Tab: “Publications”

Thanks!

And, once again, a special thanks to Rich for organizing this wonderful on-going event!