LING3701/PSYCH3371: Lecture Notes 8
A Model of Ambiguity in Sentence Processing

We have seen how complex ideas can be encoded and decoded into sentences.

This lecture will describe how this process controls for ambiguity.

Contents

8.1 Ambiguity as superposition [Smolensky, 1990, Rasmussen & Schuler, 2018]

If multiple targets are associated with the same cue vector:

$$\overbrace{\begin{bmatrix} .58 \\ .0 \\ .49 \\ .0 \\ .0 \\ .67 \\ .0 \end{bmatrix}}^{\text{target 1 (insect)}} \overbrace{\begin{bmatrix} .0 & .58 & .0 & .0 & .58 & .0 & .58 \end{bmatrix}}^{\text{cue (e.g. the word `}bug\text{')}} + \overbrace{\begin{bmatrix} .0 \\ .50 \\ .50 \\ .0 \\ .50 \\ .0 \\ .50 \end{bmatrix}}^{\text{target 2 (microphone)}} \overbrace{\begin{bmatrix} .0 & .58 & .0 & .0 & .58 & .0 & .58 \end{bmatrix}}^{\text{cue (e.g. the word `}bug\text{')}}$$

$$= \begin{bmatrix} .0 & .33 & .0 & .0 & .33 & .0 & .33 \\ .0 & .0 & .0 & .0 & .0 & .0 & .0 \\ .0 & .28 & .0 & .0 & .28 & .0 & .28 \\ .0 & .0 & .0 & .0 & .0 & .0 & .0 \\ .0 & .0 & .0 & .0 & .0 & .0 & .0 \\ .0 & .38 & .0 & .0 & .38 & .0 & .38 \\ .0 & .0 & .0 & .0 & .0 & .0 & .0 \end{bmatrix} + \begin{bmatrix} .0 & .0 & .0 & .0 & .0 & .0 & .0 \\ .0 & .29 & .0 & .0 & .29 & .0 & .29 \\ .0 & .29 & .0 & .0 & .29 & .0 & .29 \\ .0 & .0 & .0 & .0 & .0 & .0 & .0 \\ .0 & .29 & .0 & .0 & .29 & .0 & .29 \\ .0 & .0 & .0 & .0 & .0 & .0 & .0 \\ .0 & .29 & .0 & .0 & .29 & .0 & .29 \end{bmatrix} = \begin{bmatrix} .0 & .33 & .0 & .0 & .33 & .0 & .33 \\ .0 & .29 & .0 & .0 & .29 & .0 & .29 \\ .0 & .57 & .0 & .0 & .57 & .0 & .57 \\ .0 & .0 & .0 & .0 & .0 & .0 & .0 \\ .0 & .29 & .0 & .0 & .29 & .0 & .29 \\ .0 & .38 & .0 & .0 & .38 & .0 & .38 \\ .0 & .29 & .0 & .0 & .29 & .0 & .29 \end{bmatrix}$$

when associative memory is cued using that vector, the result is both vectors, **superposed**:

$$\overbrace{\begin{bmatrix}.58\\.0\\.49\\.0\\.0\\.67\\.0\end{bmatrix}}^{\text{target 1 (insect)}} + \overbrace{\begin{bmatrix}.0\\.50\\.50\\.0\\.50\\.0\\.50\end{bmatrix}}^{\text{target 2 (microphone)}} = \overbrace{\begin{bmatrix}.58\\.50\\.99\\.0\\.50\\.67\\.50\end{bmatrix}}^{\text{combined target}} = \overbrace{\begin{bmatrix}.0 & .33 & .0 & .0 & .33 & .0 & .33\\ .0 & .29 & .0 & .0 & .29 & .0 & .29\\ .0 & .57 & .0 & .0 & .57 & .0 & .57\\ .0 & .0 & .0 & .0 & .0 & .0 & .0\\ .0 & .29 & .0 & .0 & .29 & .0 & .29\\ .0 & .38 & .0 & .0 & .38 & .0 & .38\\ .0 & .29 & .0 & .0 & .29 & .0 & .29\end{bmatrix}}^{\text{synaptic weights}} \overbrace{\begin{bmatrix}.0\\.58\\.0\\.0\\.58\\.0\\.58\end{bmatrix}}^{\text{cue ('}bug\text{')}}$$

<u>PRACTICE:</u> If associative memory M is made from one cue u and two targets v_1 and v_2, as below:

$$M = \overbrace{\begin{bmatrix}.0\\.0\\.80\\.0\\.0\\.0\\.60\end{bmatrix}}^{\text{target } v_1 \text{ (lender)}} \overbrace{\begin{bmatrix}.0 & .50 & .50 & .50 & .0 & .50 & .0\end{bmatrix}}^{\text{cue } u \text{ ('}bank\text{')}} + \overbrace{\begin{bmatrix}.58\\.58\\.0\\.0\\.58\\.0\\.0\end{bmatrix}}^{\text{target } v_2 \text{ (shore)}} \overbrace{\begin{bmatrix}.0 & .50 & .50 & .50 & .0 & .50 & .0\end{bmatrix}}^{\text{cue } u \text{ ('}bank\text{')}}$$

what is the result of cueing M with u? (HINT: you don't have to calculate the matrix!)

$$\overbrace{\begin{bmatrix}\ \\ \ \\ \ \\ \ \\ \ \\ \ \\ \ \end{bmatrix}}^{\text{result}} = \overbrace{\begin{bmatrix} & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \end{bmatrix}}^{\text{synaptic weights } M} \overbrace{\begin{bmatrix}.0\\.50\\.50\\.50\\.0\\.50\\.0\end{bmatrix}}^{\text{cue } u \text{ ('}bank\text{')}}$$

Describe the result in terms of v_1 and v_2.

8.2 Propagation of ambiguity

If we take a (non-interfering) set of associations:

$$\overbrace{\begin{bmatrix}.58\\.0\\.58\\.0\\.0\\.58\\.0\end{bmatrix}}^{\text{target 1 (sting)}}\overbrace{\begin{bmatrix}.0 & .58 & .0 & .0 & .58 & .0 & .58\end{bmatrix}}^{\text{cue 1 (insect)}} + \overbrace{\begin{bmatrix}.0\\.50\\.50\\.0\\.50\\.0\\.50\end{bmatrix}}^{\text{target 2 (record)}}\overbrace{\begin{bmatrix}.0 & .0 & .58 & .58 & .0 & .58 & .0\end{bmatrix}}^{\text{cue 2 (microphone)}}$$

$$= \begin{bmatrix}.0 & .33 & .0 & .0 & .33 & .0 & .33\\ .0 & .0 & .0 & .0 & .0 & .0 & .0\\ .0 & .33 & .0 & .0 & .33 & .0 & .33\\ .0 & .0 & .0 & .0 & .0 & .0 & .0\\ .0 & .0 & .0 & .0 & .0 & .0 & .0\\ .0 & .33 & .0 & .0 & .33 & .0 & .33\\ .0 & .0 & .0 & .0 & .0 & .0 & .0\end{bmatrix} + \begin{bmatrix}.0 & .0 & .0 & .0 & .0 & .0 & .0\\ .0 & .0 & .29 & .29 & .0 & .29 & .0\\ .0 & .0 & .29 & .29 & .0 & .29 & .0\\ .0 & .0 & .0 & .0 & .0 & .0 & .0\\ .0 & .0 & .29 & .29 & .0 & .29 & .0\\ .0 & .0 & .0 & .0 & .0 & .0 & .0\\ .0 & .0 & .29 & .29 & .0 & .29 & .0\end{bmatrix} = \begin{bmatrix}.0 & .33 & .0 & .0 & .33 & .0 & .33\\ .0 & .0 & .29 & .29 & .0 & .29 & .0\\ .0 & .33 & .29 & .29 & .33 & .29 & .33\\ .0 & .0 & .0 & .0 & .0 & .0 & .0\\ .0 & .0 & .29 & .29 & .0 & .29 & .0\\ .0 & .33 & .0 & .0 & .33 & .0 & .33\\ .0 & .0 & .29 & .29 & .0 & .29 & .0\end{bmatrix}$$

and cue them with a combination of states, we get a proportional combination of targets:

$$\overbrace{\begin{bmatrix}.29\\.0\\.29\\.0\\.0\\.29\\.0\end{bmatrix}}^{\text{target 1 (sting)}} + \overbrace{\begin{bmatrix}.0\\.50\\.50\\.0\\.50\\.0\\.50\end{bmatrix}}^{\text{target 2 (record)}} = \overbrace{\begin{bmatrix}.29\\.50\\.79\\.0\\.50\\.29\\.50\end{bmatrix}}^{\text{combined target}} = \overbrace{\begin{bmatrix}.0 & .33 & .0 & .0 & .33 & .0 & .33\\ .0 & .0 & .29 & .29 & .0 & .29 & .0\\ .0 & .33 & .29 & .29 & .33 & .29 & .33\\ .0 & .0 & .0 & .0 & .0 & .0 & .0\\ .0 & .0 & .29 & .29 & .0 & .29 & .0\\ .0 & .33 & .0 & .0 & .33 & .0 & .33\\ .0 & .0 & .29 & .29 & .0 & .29 & .0\end{bmatrix}}^{\text{synaptic weights}} \overbrace{\begin{bmatrix}.0\\.29\\.58\\.58\\.29\\.58\\.29\end{bmatrix}}^{\text{combined cue (insect + microphone)}}$$

This is important because it allows ambiguity to propagate through a mental process.

PRACTICE: If associative memory M is made from cues u_1 and u_2 and targets v_1 and v_2, as below:

$$M = \overbrace{\begin{bmatrix}.0\\.50\\.50\\.0\\.0\\.50\\.50\end{bmatrix}}^{\text{target } v_1 \text{ (cheap)}} \overbrace{\begin{bmatrix}.0 & .0 & .0 & .80 & .0 & .0 & .60\end{bmatrix}}^{\text{cue } u_1 \text{ (lender)}} + \overbrace{\begin{bmatrix}.80\\.0\\.0\\.0\\.60\\.0\\.0\end{bmatrix}}^{\text{target } v_2 \text{ (stony)}} \overbrace{\begin{bmatrix}.50 & .50 & .50 & .0 & .30 & .40 & .0\end{bmatrix}}^{\text{cue } u_2 \text{ (shore)}}$$

what is the result of cueing M with a mixture of $.7u_1$ and $.3u_2$? (HINT: don't calculate the matrix!)

$$\overbrace{\begin{bmatrix}\ \\ \ \\ \ \\ \ \\ \ \\ \ \\ \ \end{bmatrix}}^{\text{result}} = \overbrace{\begin{bmatrix} & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \end{bmatrix}}^{\text{synaptic weights } M} \overbrace{\begin{bmatrix}.15\\.15\\.15\\.56\\.09\\.12\\.42\end{bmatrix}}^{.7u_1+.3u_2 \text{ combined cue}}$$

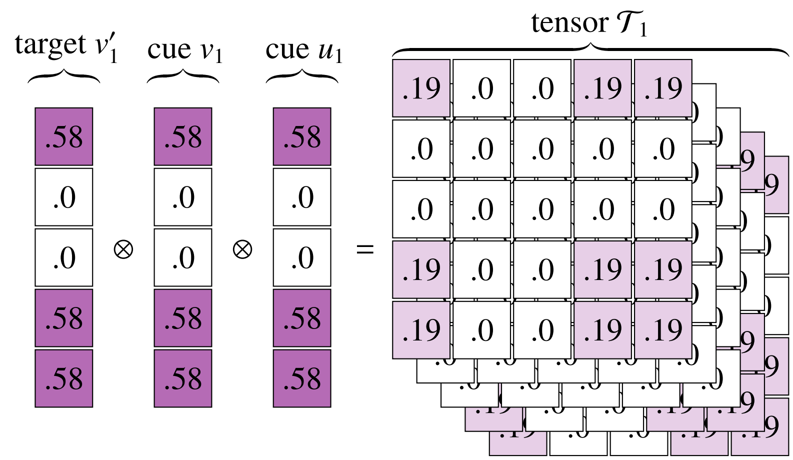
Describe the result in terms of v_1 and v_2.

8.3 Resolution of ambiguity

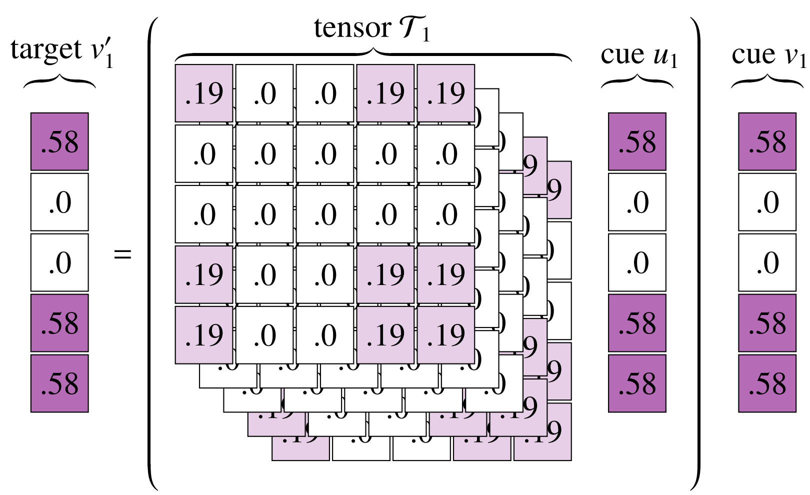
Recall the outer product of two vectors produces a matrix with pointwise products:

$$\overbrace{\begin{bmatrix}.0\\.71\\.0\\.0\\.71\end{bmatrix}}^{\text{target } v_1} \overbrace{\begin{bmatrix}.0 & .71 & .0 & .0 & .71\end{bmatrix}}^{\text{cue } u_1} = \overbrace{\begin{bmatrix}.0 & .0 & .0 & .0 & .0\\ .0 & .50 & .0 & .0 & .50\\ .0 & .0 & .0 & .0 & .0\\ .0 & .0 & .0 & .0 & .0\\ .0 & .50 & .0 & .0 & .50\end{bmatrix}}^{\text{associations } M_1} \tag{1}$$

This generalizes to triples of vectors as a **tensor product**:



Targets are then cued by multiplication (left-associative):

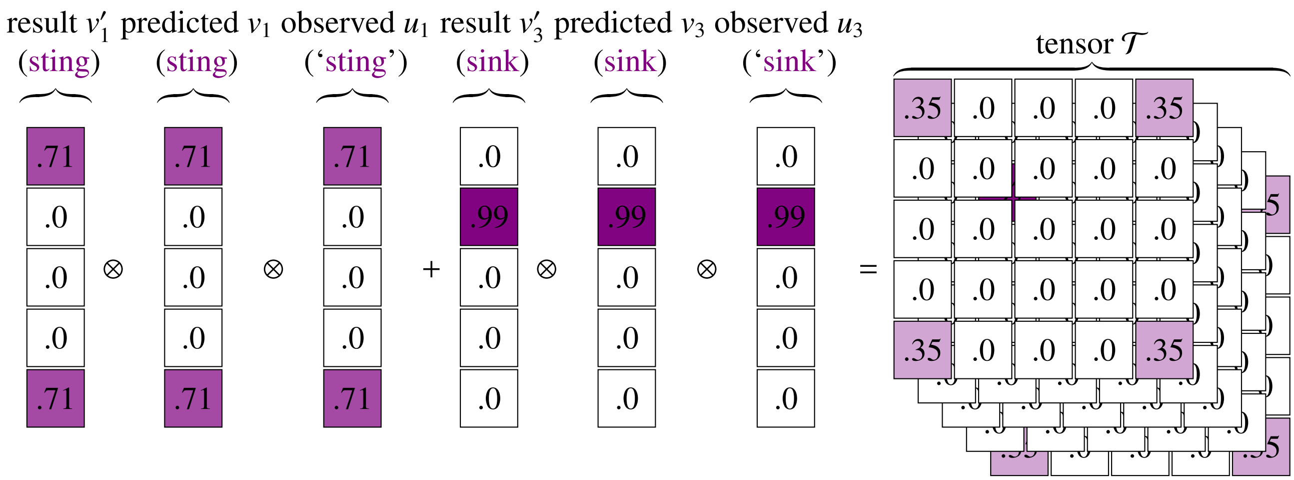


$$\underbrace{\begin{bmatrix} .58 \\ .0 \\ .0 \\ .58 \\ .58 \end{bmatrix}}_{\text{target } v_1'} = \underbrace{\begin{bmatrix} .33 & .0 & .0 & .33 & .33 \\ .0 & .0 & .0 & .0 & .0 \\ .0 & .0 & .0 & .0 & .0 \\ .33 & .0 & .0 & .33 & .33 \\ .33 & .0 & .0 & .33 & .33 \end{bmatrix}}_{\text{matrix } \mathcal{T}_1\, u_1} \underbrace{\begin{bmatrix} .58 \\ .0 \\ .0 \\ .58 \\ .58 \end{bmatrix}}_{\text{cue } v_1}$$

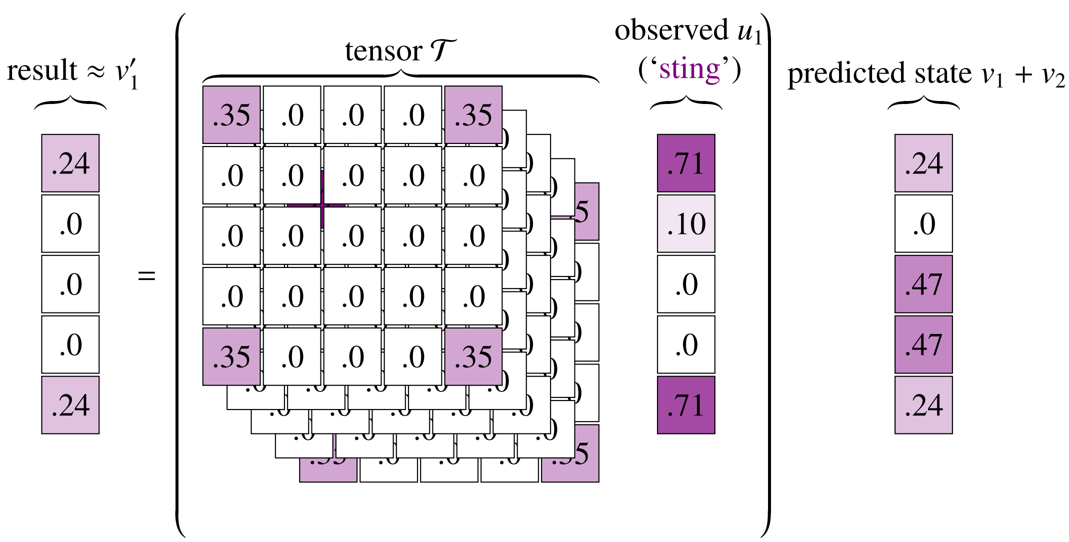
This can be implemented with ‘switched’ connections.

This gives us a means to combine ‘top-down’ predictions with ‘bottom-up’ observations…

Build auto-associations of all states:



Then cue on observed state to pick out compatible component of mixed source state:



Note the magnitude of the target is reduced compared to the source.

This reduction correlates with reading time delays (‘surprisal’) on encountering unpredicted words.

(It may take time proportional to the reduction to ‘amp up’ this state to unit magnitude.)

PRACTICE: If a filter F is made from auto-associated vectors v_2 and v_3, as below:

$$F = \overbrace{\begin{bmatrix} .80 \\ .0 \\ .0 \\ .0 \\ .60 \\ .0 \\ .0 \end{bmatrix}}^{v_2\text{ (stony)}} \overbrace{\begin{bmatrix} .80 & .0 & .0 & .0 & .60 & .0 & .0 \end{bmatrix}}^{v_2\text{ (stony)}} + \overbrace{\begin{bmatrix} .0 \\ .58 \\ .58 \\ .0 \\ .0 \\ .0 \\ .58 \end{bmatrix}}^{v_3\text{ (phony)}} \overbrace{\begin{bmatrix} .0 & .58 & .58 & .0 & .0 & .0 & .58 \end{bmatrix}}^{v_3\text{ (phony)}}$$

what is the result of cueing F with a mixture of $.8v_1$ and $.2v_2$? (HINT: don't calculate the matrix!)

$$\overbrace{\begin{bmatrix} \\ \\ \\ \\ \\ \\ \\ \end{bmatrix}}^{\text{result}} = \overbrace{\begin{bmatrix} & & & & & & \\ &&&&&& \\ &&&&&& \\ &&&&&& \\ &&&&&& \\ &&&&&& \\ &&&&&& \end{bmatrix}}^{\text{filter } F} \overbrace{\begin{bmatrix} .16 \\ .0 \\ .0 \\ .48 \\ .12 \\ .64 \\ .0 \end{bmatrix}}^{.8v_1+.2v_2 \text{ combined cue}}$$

8.4 Probabilistic disambiguation

This multiplicative disambiguation is important because it allows probabilistic reasoning.

If:

- the prediction vector represents a **prior probability** of an idea $\mathsf{P}(i)$, and
- the filter matrix represents a **likelihood** $\mathsf{P}(w \mid i)$ of a word w given an idea i,

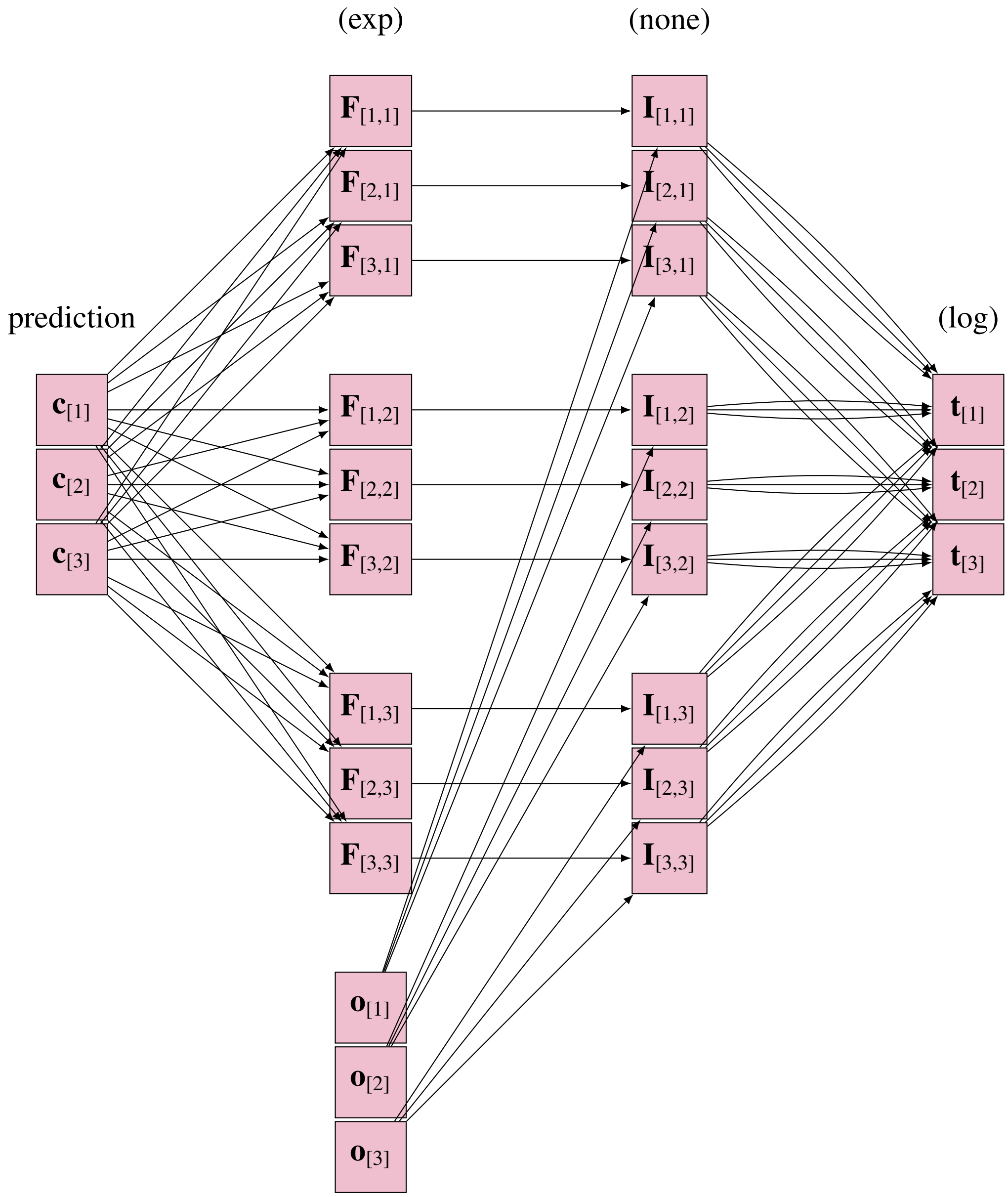
then:

- the tensor model will calculate a correct **posterior probability** of the idea given the word!

$$\mathsf{P}(i \mid w) = \frac{\mathsf{P}(i) \cdot \mathsf{P}(w \mid i)}{\mathsf{P}(w)} \quad \text{(rescaled by a constant } \mathsf{P}(w) \text{ to sum to one)}$$

8.5 Neural ambiguity resolution

Here's a (hypothetical) neural implementation of tensor disambiguation:



1. prediction **c** is weighted by tensor weights to give filter **F** with exponential transfer function.
2. filter **F** is multiplied by observation **o** (adding exponentials) to give intermediate values **I**.
3. intermediate values **I** are log transformed and added to get filtered prediction **t**.

But any neural network can behave probabilistically if trained 'autoregressively' (to predict words).

References

[Rasmussen & Schuler, 2018] Rasmussen, N. E. & Schuler, W. (2018). Left-corner parsing with distributed associative memory produces surprisal and locality effects. *Cognitive Science*, 42(S4), 1009–1042.

[Smolensky, 1990] Smolensky, P. (1990). Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial intelligence*, 46(1-2), 159–216.