Investigating forgetting curves with learning rule-derived interferences

Yu-Hsiang Tseng (Sean) R.Harald Baayen

Department of Linguistics, University of Tübingen

2024/07/23 @ The 31st Annual ACT-R Workshop
Tilburg, the Netherland

What I would like to share

- Widrow-Hoff learning rule, forgetting curve, and the interferences.
- Widrow-Hoff learning rule is an extremely simple model; it is purely based on inputs stimuli and learning targets.
- Interestingly, by continuously probing the model state of items, we could trace out something visually similar to a forgetting curve, and define an interference index from the curve.
- We apply the idea to model a fact learning dataset, where the presentation of items is optimized by SlimStampen algorithm.
- From the statistical results, it seems that these two models agree with each other.

Mappings vectors together

- We can model the comprehension as learning a mapping (F) from cue vectors (C) to semantic vectors (S).
- For example, we can use a fixed-length spectrogram as cue vectors, and learn a mapping (F) to the semantic embeddings (S).
    - Homophones do not sound the same.
- The mapping can be solved analytical with linear algebra, or use incremental learning rules.

$$\underset{(N\times d)}{C} \cdot \underset{(d\times h)}{F} = \underset{(N\times h)}{S}$$

$$\begin{bmatrix} — & c_1 & — \\ — & c_2 & — \\ & \vdots & \\ — & c_N & — \end{bmatrix} \cdot F = \begin{bmatrix} — & s_1 & — \\ — & s_2 & — \\ & \vdots & \\ — & s_N & — \end{bmatrix}$$

$$F = (C^\top C)^{-1} C^\top S$$

General form of Discriminative Lexicon Model (DLM) and the end-state solution

Incremental learning rules

$$[\dots \quad c_i \quad \dots] \cdot \begin{bmatrix} w_{11} & \cdots & \cdots \\ \vdots & w_{ij} & \\ \vdots & & \ddots \end{bmatrix} = [\dots \quad s_j \quad \dots]$$

- Learn to map a single cue vector to a semantic vector one at a time
- We can use the backpropagation to find the gradients for the model having a two-layer FFN and using mean-squared error as loss. The gradient is the Widrow-Hoff learning rule.
- If the input and output are all binary codings, it can be further simplified to Rescorla-Wagner learning rule.

$$\mathcal{L} = \frac{1}{2}\sum_{j=1}^{h}(\hat{s_j} - s_j)^2$$

$$\frac{\partial \mathcal{L}}{\partial w_{ij}} = \frac{\partial \mathcal{L}}{\partial \hat{s}_j} \cdot \frac{\partial \hat{s}_j}{w_{ij}}$$

$$= (\hat{s}_j - s_j)c_i$$

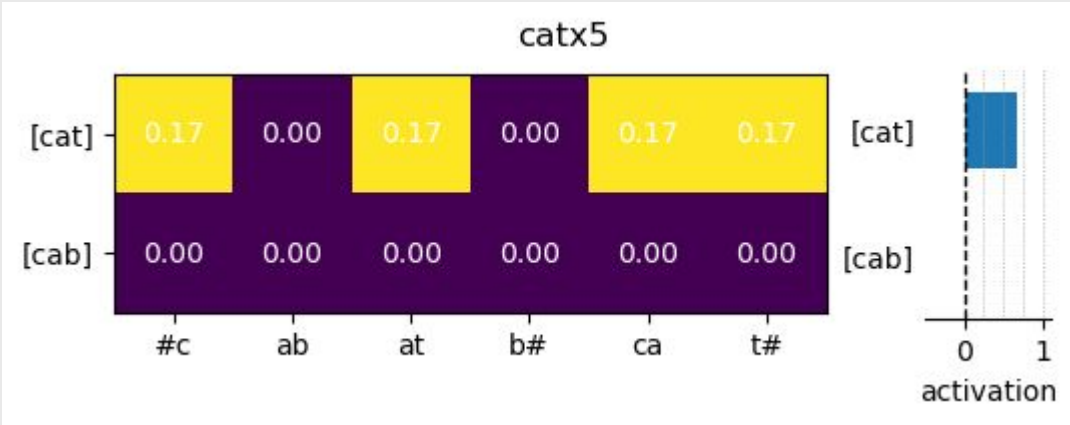
$$\text{WH: } \Delta w_{ij} = \alpha(o_j - \hat{o}_j)c_i$$

RW

$$\Delta w_{ij}^t = \begin{cases} 0 & \text{if } \textsc{absent}(C_i, t) \\ \alpha\left(\lambda - \sum_{\textsc{present}(C_k,t)} w_{kj}\right) & \text{if } \textsc{present}(C_i, t) \ \&\ \textsc{present}(O_j, t) \\ \alpha\left(0 - \sum_{\textsc{present}(C_k,t)} w_{kj}\right) & \text{if } \textsc{present}(C_i, t) \ \&\ \textsc{absent}(O_j, t) \end{cases}$$

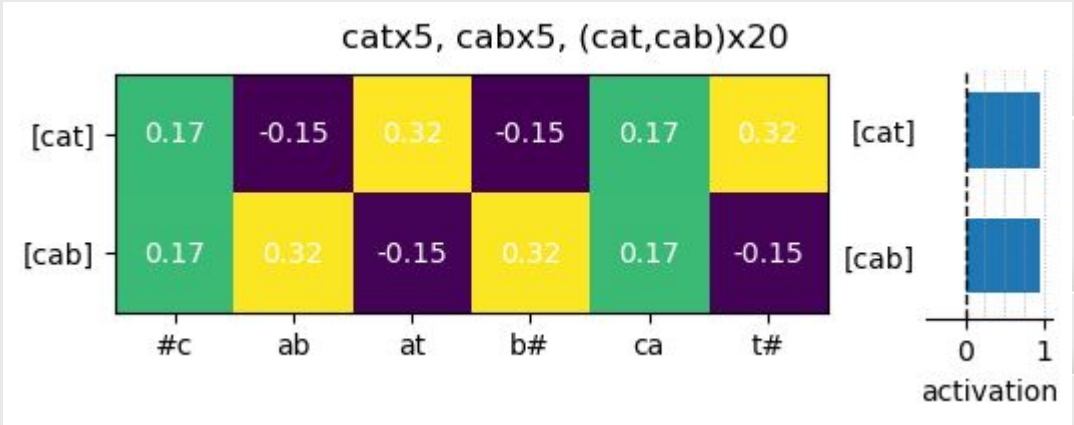
What does RW learning look like?

- Consider the form-meaning mappings of 2 words: cat and cab
- Using bigram features to encode the "form": e.g. cat is `/#c,ca,at,t#/` (# indicates word boundary), coded as `[1,0,1,0,1,1]`
- The "meanings" of cat is the binary coding of [cat] (`[1,0]`); and cab is [cab] (`[0,1]`)



Interferences from shared features

- In this example, cat, and cab shares two features: `/#c,ca/`
- These two features are competing cues:
  - In cab trial, the `/ca/` feature will wrongly activate the [cat]
- During learning, the network has to suppress the competing cues which in turn reduces the activation of the competing word => interference



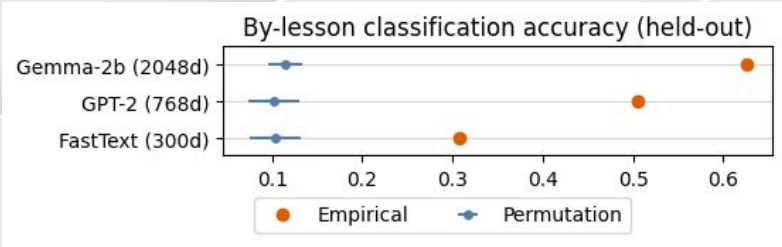
Apply the learning rule to real data

- The fact learning dataset. (thanks to Hedderik, Maarten & Thomas!)
- Participants are learning cognitive psychology terms.
  - Hypothetical units in a recognition system that respond, or fire, whenever a specific letter pair is in view. => Bigram Detectors
  - Users type in the terms, the RTs are recorded
  - SlimStampen algorithm estimates items' activations, based on which it optimizes when and how many times an item should be presented.
  - Already in the format of a series of cue-semantic mapping learning events.
- We select a subset of the dataset, which contains 12,914 trials
  - 199 sessions across 109 participants.
  - 243 unique items coming from 10 chapters in cogpsy textbooks.

How to code the texts to vectors

- The cue side is encoded by the bi-/trigram features of the definition.
  Hypothetical units => #H, Hy, yp, … #Hy, Hyp, ypo, … (3396 dimensions)
- Instead of binary coding, the semantic side, the answer (e.g. bigram detectors), is coded as a 2,048d vectors with an LLM (google/gemma-2b)

- [suppl.] the static/contextualized embeddings (CEs)
  - Static embedding (FastText): average of "bigram" and "Detectors"
  - CEs (GPT-2/Gemma-2b): Input sentence: Hypothetical units in a […].
    Bigram Detectors
  - We chose Gemma-2b, as it best
    Reflects the by-lesson structure.

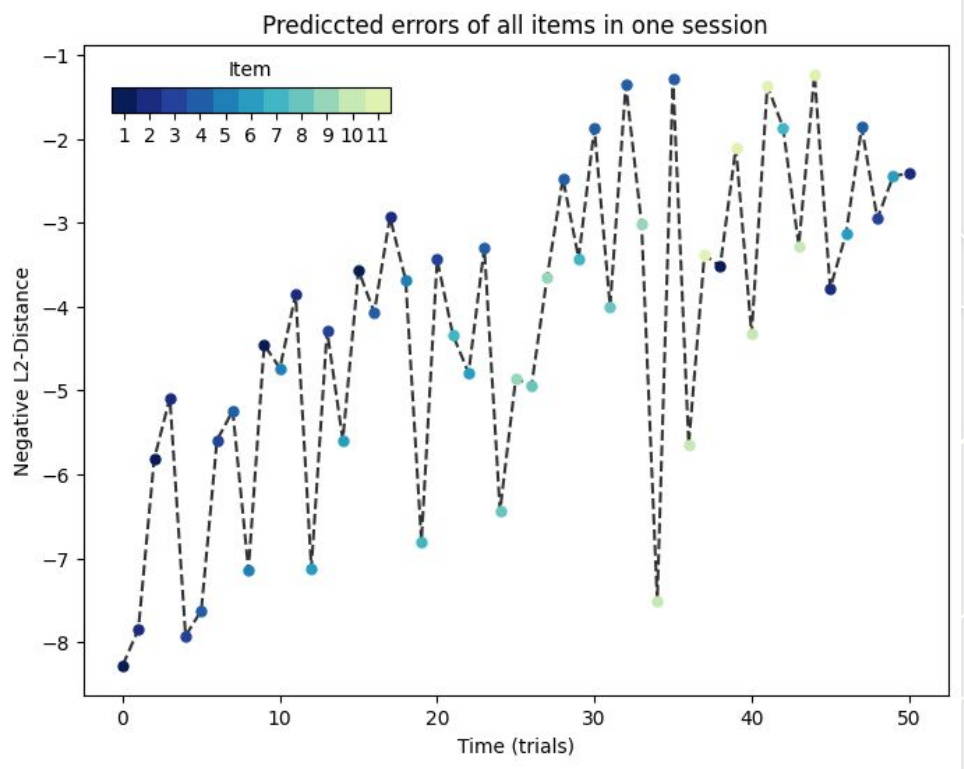


The research questions

- How does the learning rule work in a real dataset?
- We now have two different approaches to the same dataset:
  - SlimStampen algorithm: informed by the participant's RTs and uses them to estimate the item's activation.
  - Widrow-Hoff learning rule (the WH model): more form-driven, maps the n-gram features to the target semantic vectors.
- Is it possible the Widrow-Hoff is estimating the same thing but from another directions?
  - Two WH-model indices: last-l2dist and interference.

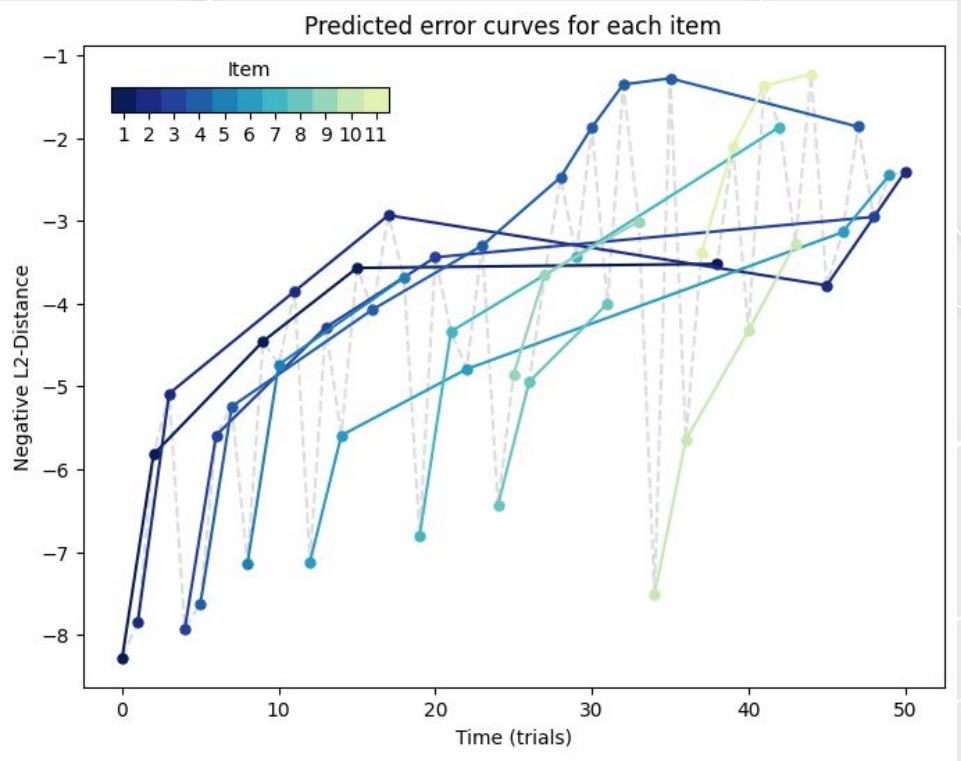
Start from by-trial prediction errors: L2-distance

- One model for each (subject-)session.
- The error (y-axis) is shown in negative L2-dist (euclidean distance, higher is better)
- The WH-model is trained incrementally, learning one trial at a time: predict, compute error, update param.



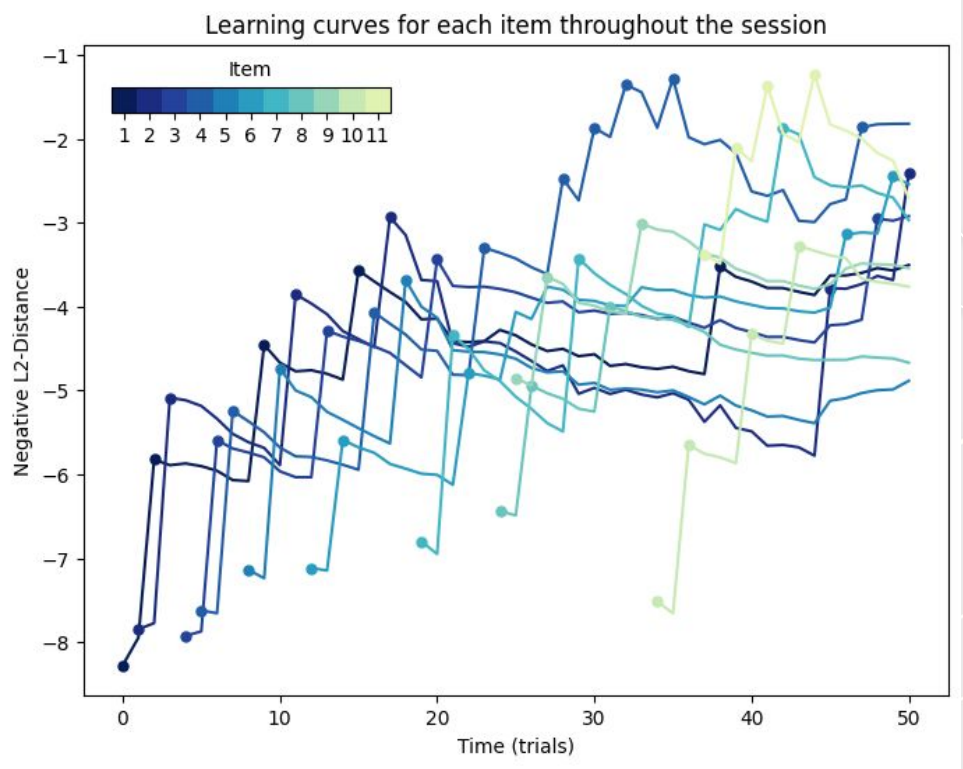
By-item curve: foreground

- Each item is introduced in different trials and repeated for multiple times.
- Connect the dots of the same item gives the learning curve for each item.
- These are when items are presented, how about when the items are not presented; when they are in "background"



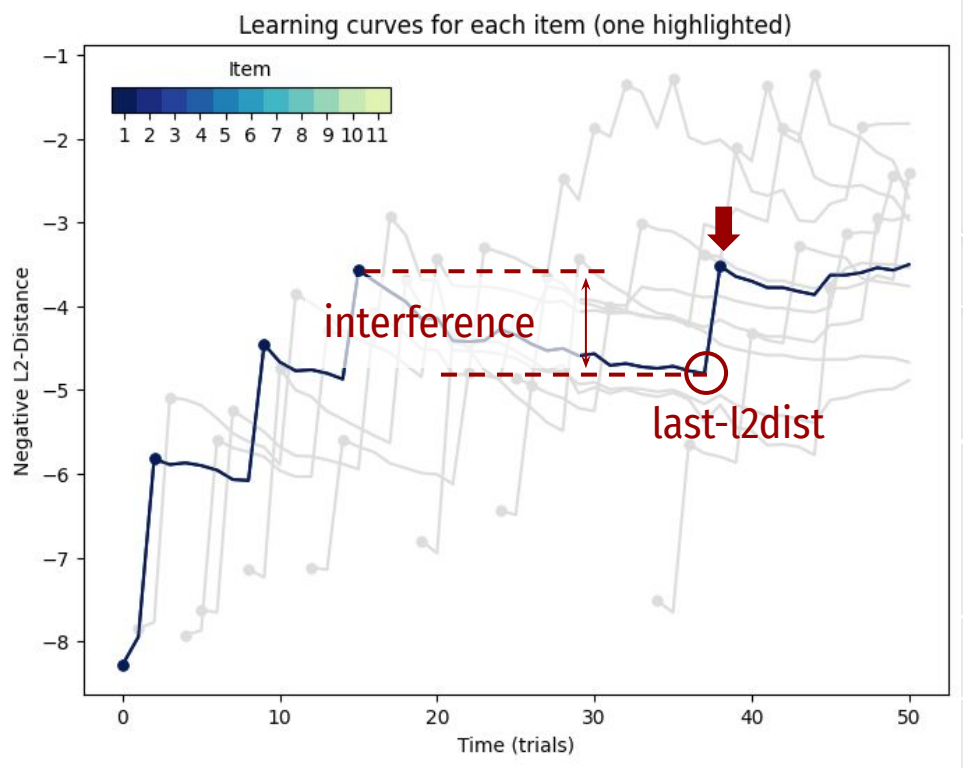
by-item curves: background

- We can run forward-pass multiple times when the item is in background, to probe how other items are doing
- But the WH-model only updates the parameters with the current learning event.
- Connect all the points of an item gives the full learning trace of that item.



Define indices from learning curves

- For each item's presentation, there are two indices:
- Last-l2dist: the euclidean distance between the predicted and true vectors, before updating
- interference: the gap between the current one and the one of last presented.
- Computed for every trials

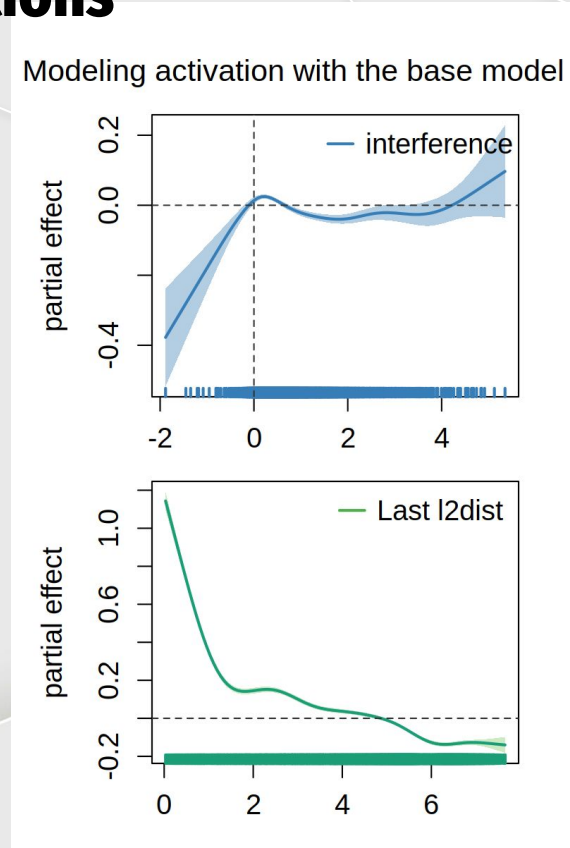


Statistical analysis

- We analyze the effects of the interference and the last_l2dist with Generalized Additive Model (GAM).
- There are two models, each with different response variables
  - SlimStampen's estimated activation
  - logRT
- Both models have the same set of predictors:
  - Interference, last_l2dist
  - Control variables: trial index, number of intervening trials between presentations, averaged duration of each intervening trial.
- All variables are modeled as smoothed effect.

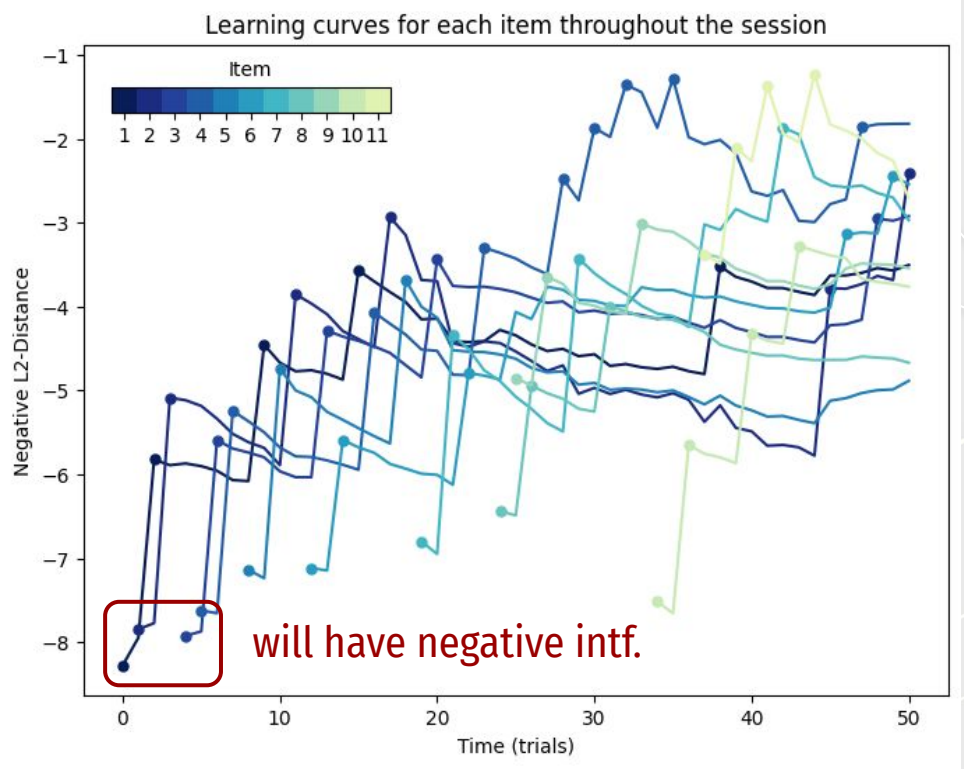
Some preliminary results of modeling activations

- The last-l2dist has a clear effect on the activation value from SlimStampen.
  - The larger the error, the lower the activation.
  - Makes sense, as it is basically the WH-model version of the activation
- The interference effect is less clear-cut:
  - For "positive interferences": generally larger the interference, the lower the activation.
  - Negative interference: warmup phase



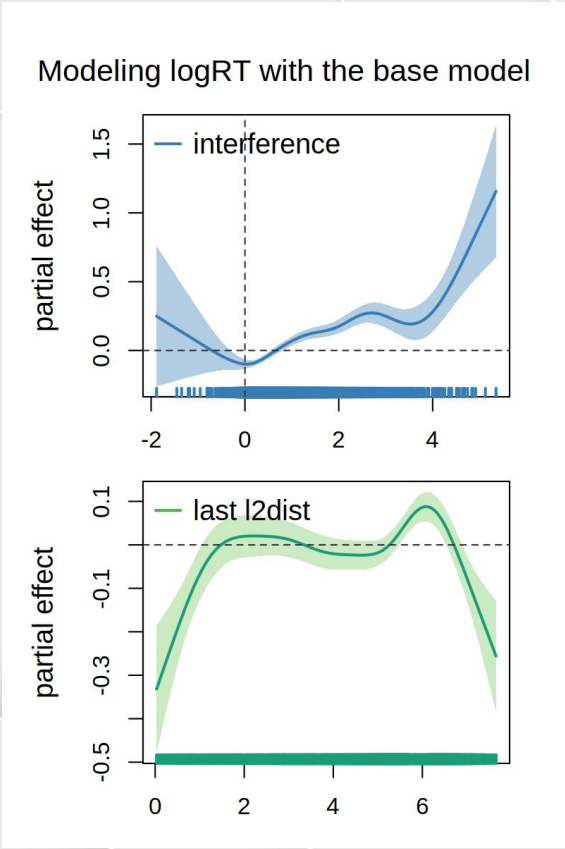
[optional] Warmup phase: the negative interference

- At very beginning of the session, the weights are starting to "align with" the target semantic vectors.
- So at the first few trials, anything helps, the l2dist will keep improving even if it comes from another items
- The interference will be negative in this case.

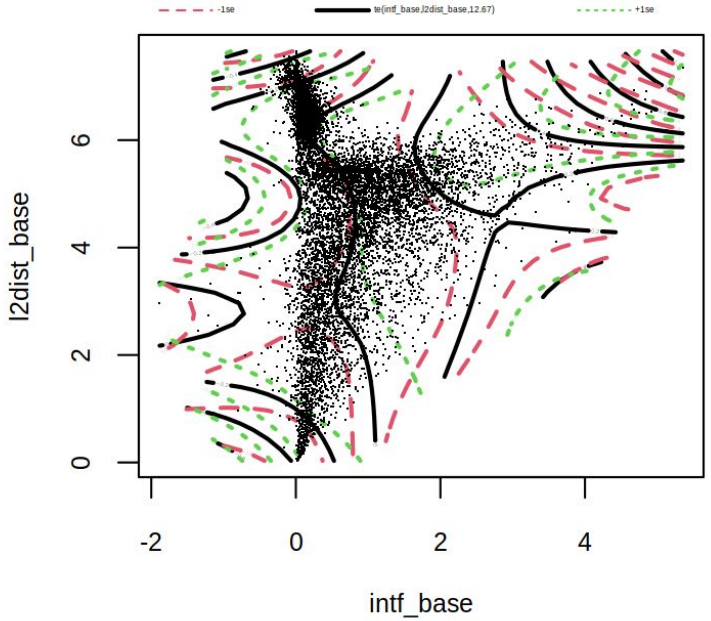


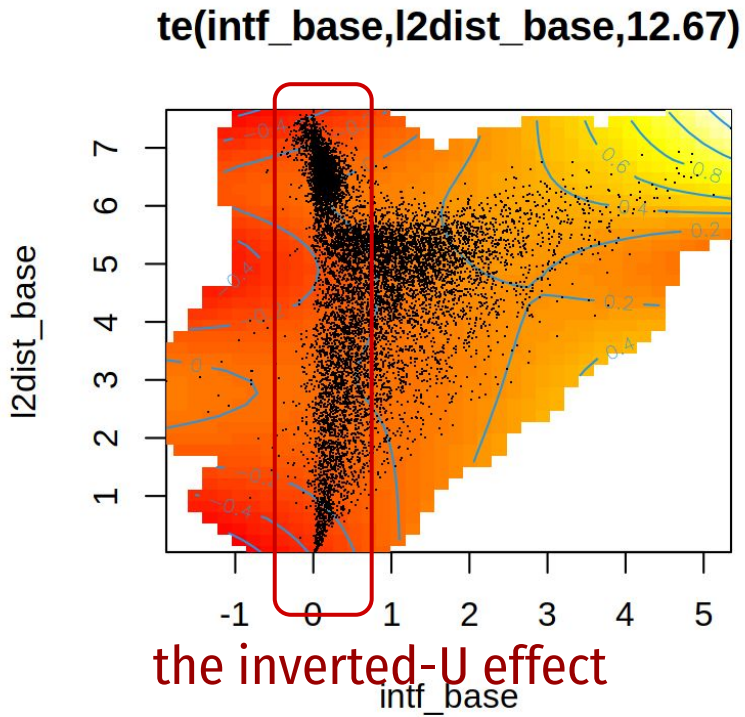
Modeling logRT

- Interference has a clear effect on logRT:
  - Larger the interference, the longer the RT
  - Has larger effect than l2dist
- Last-l2dist has an inverted U-shape pattern:
  - The first half is expected: the larger the error, the longer the RT
  - The second half is maybe related to the newly introduced items: they are often close together thus low interference and high l2dist. (see the tensor product effect)



[optional] logRT model: interaction of intf & l2dist

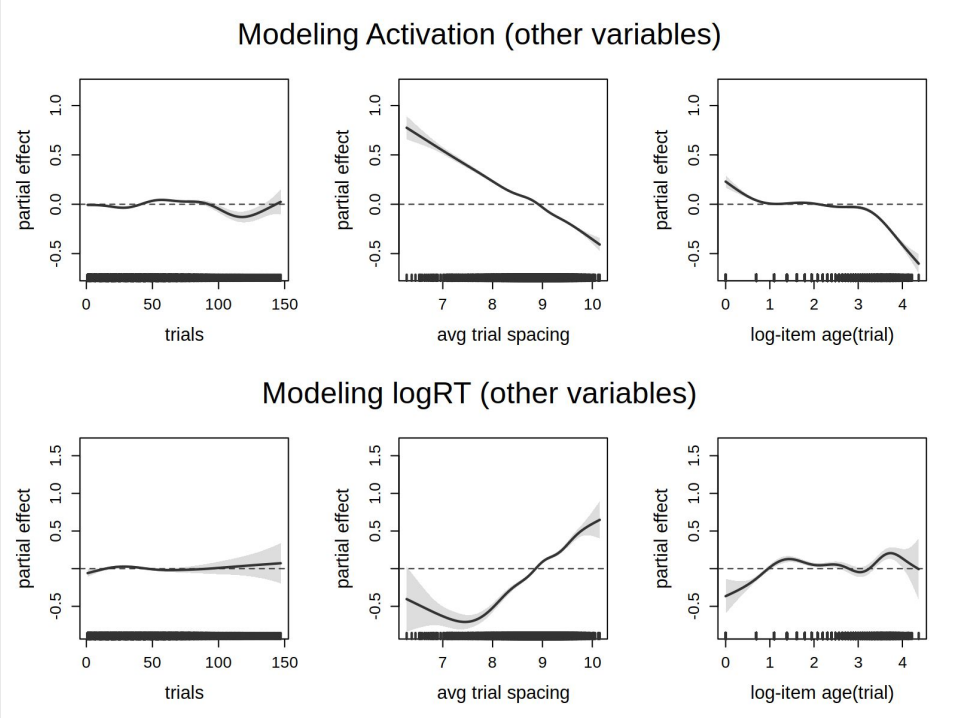




the inverted-U effect

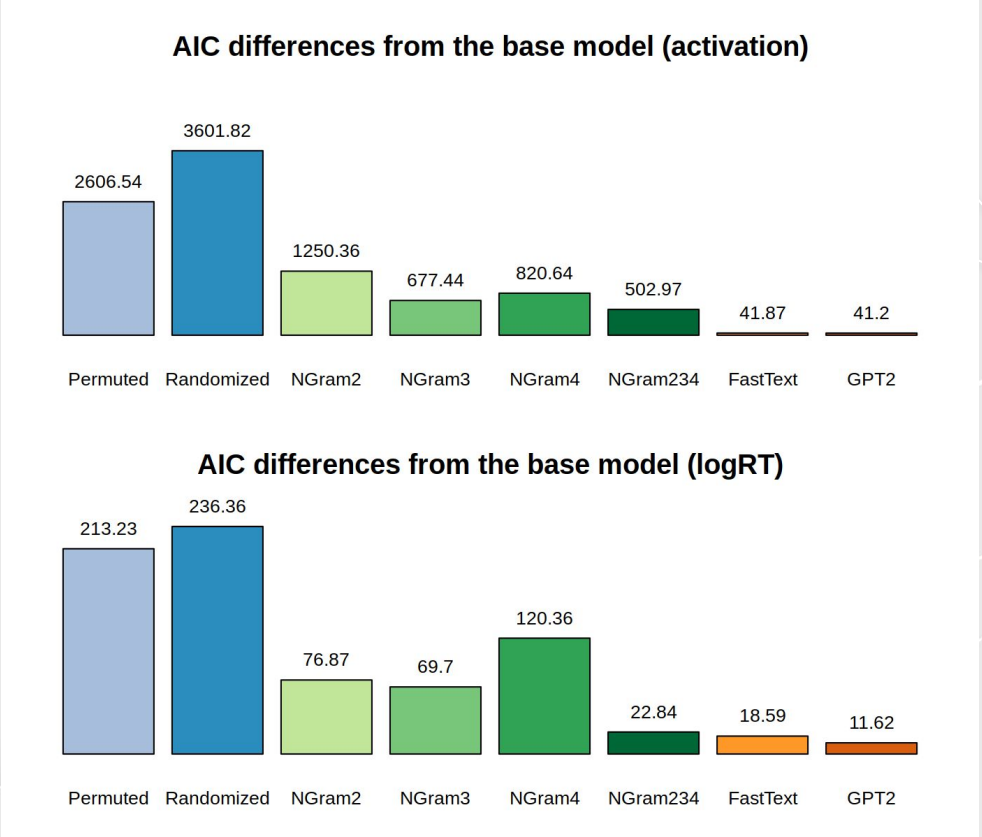
Other control variables

- trial numbers: the order of presentation
- Average trial spacing: the (log-) averaged duration of each intervening trial (act↓, RT↑)
- log-item age: number of intervening trials between presentations (act↓, RT↑)



What matters the most to the WH-model

- A sanity check: permutation and random vector makes it a lot worse. (as it should be)
- Bi/trigram we are using performs better than others
- The choice of semantic vector (FastText, GPT2, Gemma2b) doesn't matter that much.



Take home message

- We use the Widrow-Hoff learning rule, we model a fact learning dataset as a cue-semantic mapping task.
- By probing the "background items," we can trace out the learning curves of each item, based on which we define two indices, interference and last-l2dist.
- We found that the last-l2dist are consistent with SlimStampen's activation estimates, and the interferences show clear effect on RTs.
- Time-related control variables still have an effect, suggesting there is something the learning-rule indices haven't account for.

Thank you!

Special thanks to Hedderik van Rijn, Maarten van der Velde & T. J. (Thomas) Wilschut

Investigating forgetting curves with learning rule-derived interferences

Yu-Hsiang Tseng (Sean) R.Harald Baayen
Department of Linguistics, University of Tübingen

2024/07/23 @ The 31st Annual ACT-R Workshop
Tilburg, the Netherland