Quantitative Comparison for Generative Theories: Embedding Competence Linguistic Theories in Cognitive Architectures and Bayesian Models

ADRIAN BRASOVEANU[†] & JAKUB DOTLAČIL[‡]
[†]*UC Santa Cruz &* [‡]*University of Amsterdam* [*]

1 Introduction

The main goal of this paper is to introduce a framework integrating generative theories, computational cognitive models, and Bayesian methods. The integration proceeds in two parts. First, competence-level generative theories are embedded in performance-level processing theories formulated in the ACT-R cognitive architecture (Adaptive Control of Thought-Rational; Anderson and Lebiere 1998, Lewis and Vasishth 2005 a.o.). Second, these integrated competence-performance processing theories become part of a Bayesian model, which can be fitted to experimental data.

The main upshot is that we are able to consider alternative generative grammar theories and quantitatively compare how well they fit experimental data. A detailed introduction to the framework will be available soon in Brasoveanu and Dotlačil (in prep.). In this paper, we focus on a case study: the lexical decision task in Murray and Forster (2004). We model their data with 3 different ACT-R models that differ qualitatively and/or quantitatively. We then fit these models to the Murray and Forster (2004) experimental data and compare the results.

Our generative grammar + ACT-R + Bayes framework is very general: it enables us to incorporate rich syntactic and semantic theories, and also model experimental tasks other than the one considered here. We choose to model lexical decision for reasons of clarity and transparency: this task is straightforward, so it makes the general structure of our approach very transparent. The integration of generative theories and ACT-R is computationally implemented in a new Python3 library *pyactr*.[1]

[*]We are grateful to Donka Farkas, Abel Rodriguez, Matt Wagers and the UCSC S-lab audience (January 2018) for comments and discussion. The usual disclaimers apply.

[1]Readers familiar with ACT-R know that the official implementation of ACT-R is in Lisp. Using *pyactr* enables us to easily interface ACT-R models with widely-used Python3 libraries for Bayesian modeling, e.g., *pymc3*. The most relevant parts of the code are provided in the main text; the full code will be available in

The paper is structured as follows. In Section §2, we introduce the lexical decision task and the data we model, and discuss a basic Bayesian log-frequency model for this data. This model highlights the imperfect data fit of the log-frequency assumption, and introduces the basic structure of Bayesian models that we will need later. In Section §3, we introduce the main idea behind our ACT-R models of lexical decision: frequency effects as practiced memory retrieval. In Section §4, we introduce a series of 3 ACT-R models of a participant completing the lexical decision task, and we quantitatively compare them. These lexical access models are particularly simple; the concluding section (§5) briefly outlines how the framework can accommodate much more realistic linguistic theories.

2 The lexical decision task and a Bayesian log-frequency model

Word frequency is one very robust parameter affecting latencies and accuracies in lexical decision tasks (Whaley, 1978). Frequency effects have been found in many, if not all tasks that involve some kind of lexical processing (Forster, 1990; Monsell, 1991). These effects are assumed to have a specific functional form: lexical access latency can be well approximated as a log-function of word frequency (Howes and Solomon 1951).

Murray and Forster (2004) studied the role of frequency in detail and identified various issues with the log-frequency model. Their data consisted of collected responses and response times in a lexical decision task using words from 16 frequency bands: these 16 word-frequency bands (measured in tokens-per-million) together with the results of Experiment 1 in Murray and Forster (2004), are provided in Table 1 below. In what follows, we will use the mean frequency listed in the 2nd column from the left as the predictor variable, and we will model the lexical decision latencies and accuracies reported in columns 3 and 4 in terms of this predictor.

To get acquainted with the structure of a Bayesian model, and as a baseline for all our future models, we specify a simple Bayesian log-frequency model for this data. The basic structure of the model is provided in Table 2. The last line in Table 2 shows that the log-frequency model takes the observed reactions times (RTs) for a word to be a linear function of the log-frequency of that word + some normal/Gaussian-distributed noise. This linear

Brasoveanu and Dotlačil (in prep.).

Table 1: Exp. 1 in Murray and Forster (2004)

Frequency range	**Mean frequency**	**Latency (ms)**	**Accuracy (%)**
315–197	**242.0**	**542**	**97.22**
100–85	**92.8**	**555**	**95.56**
60–55	**57.7**	**566**	**95.56**
42–39	**40.5**	**562**	**96.3**
32–30	**30.6**	**570**	**96.11**
24–23	**23.4**	**569**	**94.26**
19	**19.0**	**577**	**95**
16	**16.0**	**587**	**92.41**
14-13	**13.4**	**592**	**91.67**
12–11	**11.5**	**605**	**93.52**
10	**10.0**	**603**	**91.85**
9	**9.0**	**575**	**93.52**
7	**7.0**	**620**	**91.48**
5	**5.0**	**607**	**90.93**
3	**3.0**	**622**	**84.44**
1	**1.0**	**674**	**74.63**

function is the *likelihood* component of our Bayesian model, which connects the model to the data.

To estimate a linear function, we need to estimate two parameters: its intercept and its slope. A Bayesian model specifies our prior beliefs about these parameters (the first three rows in Table 2), which can be very vague and unconstrained, e.g., we can take them to be normally distributed with a mean of 0 and a large standard deviation, e.g., 300 ms.[2] Furthermore, we also specify a half-normal prior distribution for the noise – half-normal because the noise parameter is a standard deviation, so it is has to be positive. The Bayesian model then updates these priors with the information provided by the data, and outputs the posterior distributions of these parameters; technically speaking, the model draws sufficiently many samples from the posterior distributions such that the resulting empirical distributions approximate the true posteriors very well (the last row in Table 2).

The full code for all the models introduced in this paper, as well as detailed discussions of related technical issues, are provided in Brasoveanu and Dotlačil (in prep.). For our

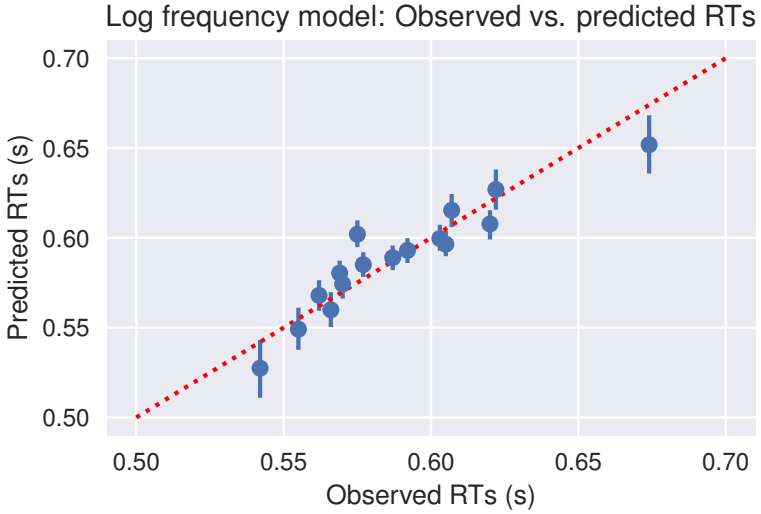
[2]The (half-)Gaussians in Table 2 are parametrized in terms of their means and variances, as is customary.

Table 2: Bayesian log-frequency model

Priors		
intercept	$\sim$	Normal(0, 300^2)
slope	$\sim$	Normal(0, 300^2)
noise	$\sim$	HalfNormal(0, 300^2)
Likelihood		
RT	$\sim$	Normal(intercept + slope $\cdot$ log(freq), noise^2)

purposes, it is enough to examine the plot in Figure 1 of the posterior predictions made by the log-frequency model. On the x axis, we plot the observed RTs (in seconds) for the 16 word frequency bands – see the blue dots, which correspond to column 3 in Table 1 above. On the y axis, we plot the mean posterior RTs in seconds (the same blue dots) and the associated 95% credible intervals (CRIs). The diagonal red line helps visualize the fit of the model to the data: the closer the blue dots are to the line, the closer the estimated RTs are to the observed data, and the better the model is at capturing the data. We see that the log-frequency model gets middle values right, but underestimates the time needed to access words in extreme frequency bands.

Figure 1: Posterior predictions of log-frequency model



3 Frequency effects as practiced memory retrieval in ACT-R

Our proposal is to model frequency effects as practiced memory retrieval: latency in memory recall is a power function of the amount of practice, and also of the time elapsed since individual instances of practice (Newell and Rosenbloom 1981, Anderson 1982, Logan 1990, Anderson et al. 1999). By practice, we simply mean the repeated presentation of an item, e.g., the repeated exposure to and/or use of a word in daily conversation.[3]

A concrete implementation of practiced memory retrieval is provided in the ACT-R cognitive architecture. The (base) activation of an item i is A_i, and it is based on the amount of time t_k elapsed since each rehearsal k of a word, and it is a power function of time because of its form t_k^{-d}. Activations contributed by individual rehearsals k (from 1 to the total number n of word rehearsals) are summed, and the total sum is log-compressed – see the formula in (1) below. The activation of an item is in turn used to compute accuracy (2) and latency (3) for retrieval processes. The free parameters associated with each formula are boxed in the equations and enumerated in parentheses.

(1) $$A_i = \log\left(\sum_{k=1}^{n} t_k^{-\boxed{d}}\right)$$ ($\boxed{d}$: decay)

(2) $$P_i = \left[1 + \exp\left(-\frac{A_i - \boxed{\tau}}{\boxed{s}}\right)\right]^{-1}$$ ($\boxed{s}$: noise, $\boxed{\tau}$: threshold)

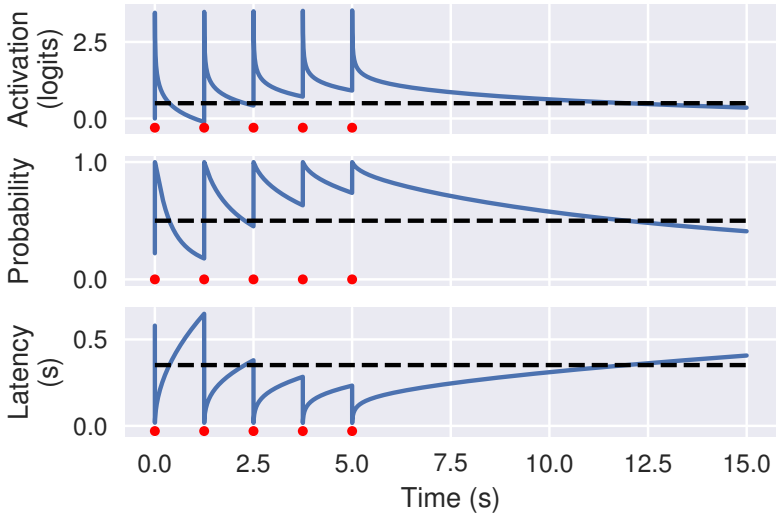
(3) $$T_i = \boxed{F} e^{-\boxed{f} A_i}$$ ($\boxed{F}$: latency factor, $\boxed{f}$: latency exponent)

As an example, Figure 2 plots activation, retrieval probability and retrieval latency as a function of time for an item presented 5 times at equally spaced 1.25 s intervals. The top plot shows that the activation of an item sharply increases after every presentation, and then decays until the next presentation. Importantly, after every presentation, the decay curve becomes shallower, ensuring that the item stays activated, i.e., above the threshold (the dotted black line), for a longer period of time. The second plots shows that the probability of successfully retrieving an item closely follows its activation. Finally, the third plot shows that latency of retrieval is inversely related to the activation of an item: the higher

[3] This proposal is different from the one in Murray and Forster (2004).

the activation of an item, the less time it takes to retrieve it.

Figure 2: Activation, retrieval probability and retrieval latency as a function of time (threshold – dotted black line; 5 presentations – red)



How do we estimate the schedule of presentations for words? For any word, the number of rehearsals that contribute to its activation are determined by its frequency (we ignore other factors throughout this paper). We generate a presentation schedule for a 15-year old speaker based on word frequency and the average number of words the 15-year old speaker is estimated to have seen (estimate based on Hart and Risley 1995; see Brasoveanu and Dotlačil in prep. for more details). With this schedule in place, we can compute activations for all 16 word-frequency bands and store them in a 16-coordinate vector we will call *activation-from-time*.

A Bayesian model for the lexical decision data is specified in Table 1, with ACT-R likelihoods for both lexical decision RTs and lexical decision accuracies. Embedding ACT-R models in Bayesian models enables us to link them to experimental data (in the case at hand: the lexical decision data from Murray and Forster 2004), and to estimate the parameters of the ACT-R models based on that data.

Table 3 shows how we specify this type of ACT-R + Bayes models. Just as in the simpler log-frequency model, we have vague, low information priors for the parameters of interest (the first five rows in Table 3). The function PYACTR-MODEL in the likelihood

Table 3: ACT-R + Bayes models of lexical decision

Priors		
d	$\sim$	Uniform(0, 1)
s	$\sim$	Uniform(0, 5)
τ	$\sim$	Normal(0, 10^2)
F	$\sim$	HalfNormal(0, 1^2)
f	$\sim$	HalfNormal(0, 1^2)
Likelihood		
μ_{RT}	$=$	PYACTR-MODEL(*activation-from-time*, d, F, f)
μ_{PROB}	$=$	$\left[1 + \exp\left(-\frac{\textit{activation-from-time} - \tau}{s}\right)\right]^{-1}$
RT	$\sim$	Normal(μ_{RT}, 0.01^2)
accuracy	$\sim$	Normal(μ_{PROB}, 0.01^2)

part invokes an ACT-R model implemented in *pyactr* and runs it to generate mean latencies μ_{RT} for the 16 word-frequency bands in Murray and Forster (2004). This function is parametrized by the activations for the 16 word-frequency bands *activation-from-time*, the decay parameter d, the latency factor F and the latency exponent f. For simplicity, we compute mean accuracies μ_{PROB} for the 16 word-frequency bands directly using equation (2), but we can also obtain them by repeated runs of the same *pyactr* model we use to obtain latencies.

In the last two lines of Table 3, the 16 observed mean RTs and accuracies from Murray and Forster (2004) are assumed to be noisy realizations of the ACT-R generated RTs and accuracies. To see if ACT-R can precisely fit the observed values, we require the normally-distributed noise to be very small in both cases.[4]

With this framework in place, we can now fit a variety of ACT-R models to data and compare their fit by specifying the models in *pyactr* and then plugging them into the likelihood function 'slot' of our Bayesian model via the PYACTR-MODEL function.

[4]See the 0.01^2 variances in the last two lines of Table 3. These small variances, as well the normal distributions, are independently motivated by the fact that we model observed *mean* RTs.

4 Three ACT-R models of lexical decision

In this section, we discuss three ACT-R models that we fit to data by embedding them in Bayesian models via the procedure shown in Table 3. ACT-R models provide the essential link to competence-level generative theories: they embed competence theories in processing models.

More specifically, for the lexical decision task we are modeling here, we have: (*i*) a symbolic competence theory of the lexicon – the structure of a lexical entry, what information is stored in it, etc.; we oversimplify here and assume lexical entries only store the written form and syntactic category of a word; and (*ii*) a symbolic performance theory of what human participants actually do in a lexical decision task – lexical items are stored in declarative memory and have an activation that is a function of their frequency, participants read a written form (sequence of characters) on the screen and attempt to retrieve a word with that form, etc.

These symbolic components are implemented in ACT-R as **condition-action** pairs (production rules) stored in procedural memory. These rules trigger a cognitive **action** if the cognitive context, i.e., the mental state of the ACT-R mind, satisfies a range of **conditions**. Depending on which production rules we use and how we formulate them, we implement different symbolic competence and performance theories in ACT-R, which can then be quantitatively compared by fitting them to the same experimental data.

The three different models we consider for the remainder of this section differ in various ways, both qualitatively (symbolically) and quantitatively (subsymbolically). For presentational simplicity, we consider symbolic and subsymbolic differences in the performance / processing hypotheses we entertain, but different competence-level representational assumptions can be implemented and compared in the exact same way.

4.1 Model 1

The first model of lexical decision we consider consists of 4 central rules. The first rule is the *attend word* rule below that takes a visual location encoded in the visual location buffer, a.k.a., the visual *where* buffer, and issues a command to the visual *what* buffer to

move attention to that visual location.[5]

(4) Rule 1 (*attend word*):

conditions				actions		
=goal>	STATE:	attend		=goal>	STATE:	retrieving
=visual-location>	ISA:	-location	⟹	+visual>	CMD:	move-attention
					SCREEN-POS:	=visual-location
?visual>	STATE:	free				

Rule 2 takes the visual value discovered at that visual location, which is a potential word form, and places a declarative memory request to retrieve a word with that form.

(5) Rule 2 (*retrieving*):

conditions				actions		
=goal>	STATE:	retrieving		=goal>	STATE:	retrieval-done
			⟹			
=visual>	VALUE:	=val		+retrieval>	ISA:	word
					FORM:	=val

Rules 3 (*lexeme retrieved*) and 4 (*no lexeme found*) take care of the two possible outcomes of the memory retrieval request: if a word with that form is retrieved from memory (*lexeme retrieved*), a command is issued to the motor module to press the *J* key, which is the Yes response; if no word is retrieved (*no lexeme found*), a command is issued to the motor module to press the *F* key, which is the No response.

(6) Rule 3 (*lexeme retrieved*):

conditions				actions		
=goal>	STATE:	retrieval-done		=goal>	STATE:	done
			⟹			
?retrieval>	BUFFER:	full		+manual>	CMD:	press-key
	STATE:	free			KEY:	J

[5]For more details about the modular structure of an ACT-R mind and the structure of the peripheral modules (visual and motor) we assume here, see Brasoveanu and Dotlačil (in prep.).

(7) Rule 4 (*no lexeme found*):

	conditions				actions	
=goal>	STATE:	retrieval-done	$\Longrightarrow$	=goal>	STATE:	done
?retrieval>	BUFFER:	empty		+manual>	CMD:	press-key
	STATE:	error			KEY:	F

Running this model with the string *elephant* as a stimulus displayed in the center of the screen, we obtain the temporal trace of the lexical-decision cognitive process in Table 4.

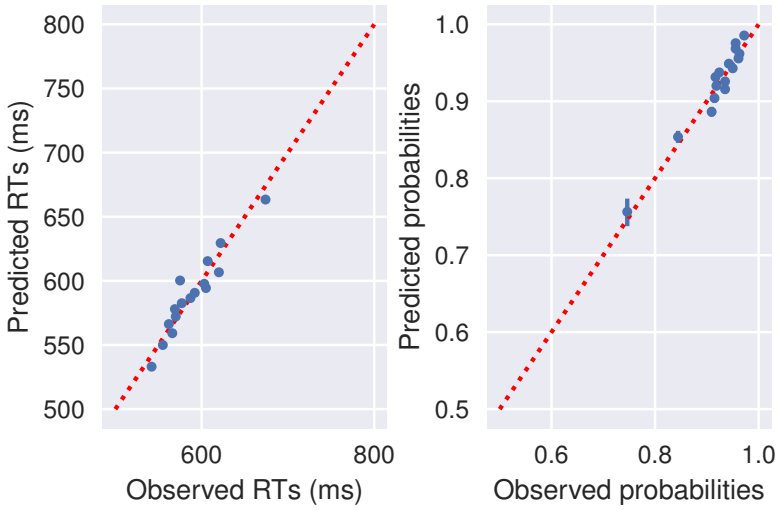
Table 4: Model 1: Temporal trace

```
****Environment: {1: {'text': 'elephant', 'position': (320, 180)}}
(0, 'PROCEDURAL', 'RULE SELECTED: attend word')
(0.05, 'PROCEDURAL', 'RULE FIRED: attend word')
(0.0679, 'PROCEDURAL', 'RULE SELECTED: retrieving')
(0.1179, 'PROCEDURAL', 'RULE FIRED: retrieving')
(0.1179, 'retrieval', 'START RETRIEVAL')
(0.1679, 'retrieval', 'RETRIEVED: word(form= elephant)')
(0.1679, 'PROCEDURAL', 'RULE SELECTED: lexeme retrieved')
(0.2179, 'PROCEDURAL', 'RULE FIRED: lexeme retrieved')
(0.2179, 'manual', 'COMMAND: press_key')
(0.4679, 'manual', 'PREPARATION COMPLETE')
(0.5179, 'manual', 'INITIATION COMPLETE')
(0.6179, 'manual', 'KEY PRESSED: J')
```

We can then take the time between the point at which a stimulus is displayed on the screen (in the 'environment') and the time at which a key is pressed as the RT that we need to fit to the experimental data from Murray and Forster (2004).[6] The posterior predictions obtained by embedding Model 1 in a Bayesian model and fitting it to data are provided in Figure 3. We see that the model fits both the latency and the accuracy data very well.

[6]We could also match the accuracy of the model (how often it presses the J key for existing words) to the Murray and Forster (2004) data by repeatedly running the *pyactr* model. As we mentioned above, for simplicity, we model accuracies directly in the Bayesian model using the ACT-R equation in (2).

Figure 3: Model 1: estimated and observed RTs and probabilities



4.2 Model 2: adding the imaginal buffer

Model 1 oversimplifies the process of encoding visually retrieved data: it assumes the visual value found at a particular visual location is immediately shuttled to the retrieval buffer. But cognition in ACT-R is *goal-driven*: any important step in a cognitive process should involve the *goal* buffer or the *imaginal* buffer, which is a goal-like buffer storing internal snapshots of the current cognitive state. In our case, it is natural to assume that the transfer between the visual and the retrieval buffer is mediated by the *imaginal* buffer.

We correct this oversimplification in a second model. The Bayesian model remains the same, the only part we change is the *pyactr*-provided likelihood for latencies. In particular, we modify the procedural core of the ACT-R model. First, we add the imaginal buffer to the model. Then, we replace the *attend word* and *retrieving* rules with three rules *attend word*, *encoding word* and *retrieving*. The new rule *encoding word* mediates between *attend word* and *retrieving*: for our limited purposes, encoding a word form means taking it from the visual buffer and shuttling it to the imaginal buffer.

(8) Rule 5 (*encoding word*):

	conditions				actions	
=goal>	STATE:	encoding	⟹	=goal>	STATE:	retrieving
=visual>	VALUE:	=val		+imaginal>	ISA:	word
					FORM:	=val

The *attend word* and *retrieving* rules are minimally revised: the output goal state for the *attend word* rule is now *encoding* (rather than *retrieving*), and the *retrieving* rule looks up the string of characters in the imaginal buffer now (rather than the visual buffer).

(9) Rule 1 (*attend word*; revised):

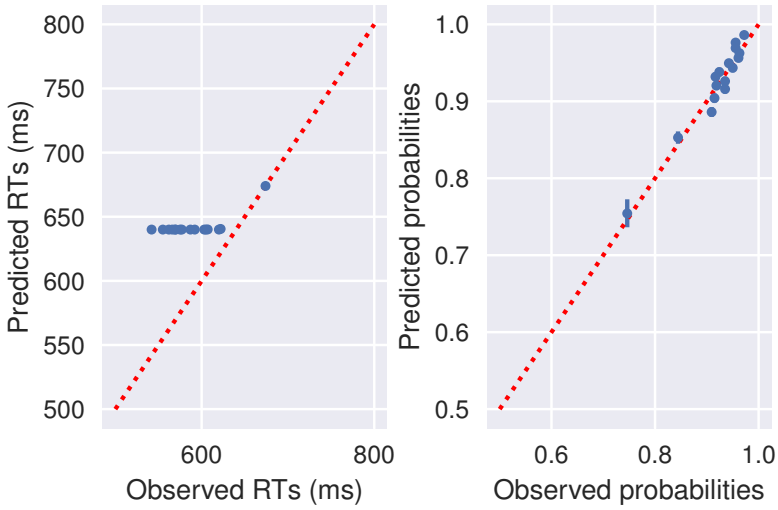
	conditions				actions	
=goal>	STATE:	attend	⟹	=goal>	STATE:	encoding
=visual-location>	ISA:	-location		+visual>	CMD:	move-attention
					SCREEN-POS:	=visual-location
?visual>	STATE:	free				

(10) Rule 2 (*retrieving*; revised):

	conditions				actions	
=goal>	STATE:	retrieving	⟹	=goal>	STATE:	retrieval-done
=imaginal>	VALUE:	=val		+retrieval>	ISA:	word
					FORM:	=val

All these modifications are symbolic / discrete / qualitative. We are nonetheless able to fit the new model to the same data and quantitatively compare its performance with Model 1 (the no-imaginal-buffer model). As the left plot in Figure 4 shows, Model 2 has a very poor fit to the latency data. The encoding step adds 200 ms to every lexical decision, since 200 ms is the default ACT-R delay for chunk-encoding into the imaginal buffer. Consequently, the predicted latencies for 15 out of the 16 word-frequency bands are greatly overestimated (above the diagonal line). Model 2 cannot run faster than about 640 ms, and this is too high to fit high-frequency words, which take about 100 ms less than that.

Figure 4: Model 2: estimated and observed RTs and probabilities



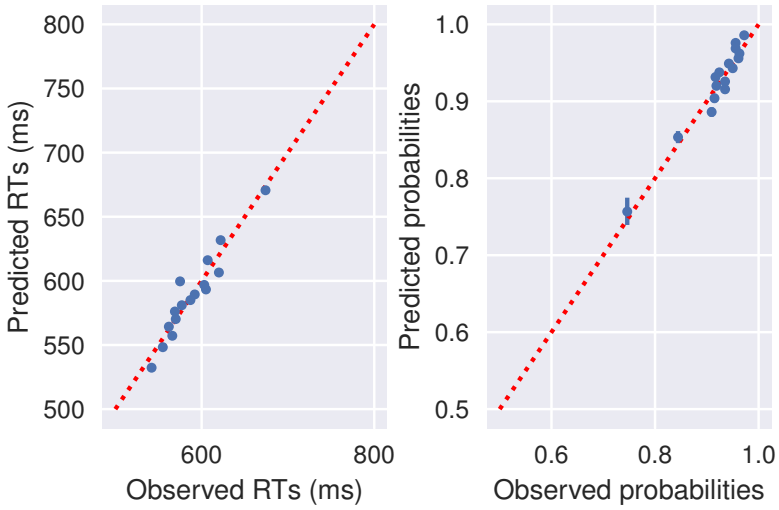
4.3 Model 3: imaginal buffer with 0 delay

Let's change a quantitative feature of Model 2 and set the imaginal delay to 0 ms instead. It is reasonable to assume that various default values for ACT-R subsymbolic parameters should be changed when modeling linguistic phenomena: natural language comprehension involves fast incremental construction of rich hierarchical representations, and this richness significantly exceeds the complexity of representations needed for other high-level cognitive processes modeled in ACT-R (e.g., arithmetic). This change is sufficient to obtain a very good fit to latencies for all 16 word-frequency bands, as shown in Figure 5.

5 Conclusion

We have presented a framework that integrates generative theories, the ACT-R cognitive architecture and Bayesian models. This framework enables us to do quantitative comparison for qualitative theories: we can implement different competence + processing models in ACT-R, and then embed these alternative ACT-R models in a Bayesian model. We can then estimate their subsymbolic parameters, and quantitatively compare these different models / theories. Consequently, we have a formally explicit way to connect competence-

Figure 5: Model 3: estimated and observed RTs and probabilities



level theories to experimental data via explicit processing models, and a formally explicit and systematic way to quantitatively compare these theories.

In this paper, we have done only informal quantitative comparisons based on posterior predictions, but systematic model comparison via WAIC values or Bayes factors is also possible – see Brasoveanu and Dotlačil (to appear) for WAIC-based model comparison. Furthermore, the framework can model eye fixation durations in eye-tracking-while-reading tasks (see Dotlačil 2018), and latencies in self-paced reading tasks targeting a variety of syntactic and semantic phenomena (Brasoveanu and Dotlačil to appear, Brasoveanu and Dotlačil in prep.).

References

Anderson, John R (1982). "Acquisition of cognitive skill". In: *Psychological review* 89.4, p. 369.

Anderson, John R. and Christian Lebiere (1998). *The Atomic Components of Thought*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Anderson, John R et al. (1999). "Practice and retention: A unifying analysis". In: *Journal of Experimental Psychology: Learning, Memory, and Cognition* 25.5, pp. 1120–1136.

Brasoveanu, Adrian and Jakub Dotlačil (in prep.). *Formal Linguistics and Cognitive Architecture*. Language, Cognition, and Mind (LCAM) Series. The *pyactr* library (Python3 ACT-R) is available here: https://github.com/jakdot/pyactr. Dordrecht: Springer.

— (to appear). "An extensible framework for mechanistic processing models: From representational linguistic theories to quantitative model comparison". In: *Proceedings of the 2018 International Conference on Cognitive Modelling*.

Dotlačil, Jakub (2018). "Building an ACT-R Reader for Eye-Tracking Corpus Data". In: *Topics in cognitive science* 10.1, pp. 144–160.

Forster, Kenneth I (1990). "Lexical processing". In: *Language: An invitation to cognitive science*. Ed. by Daniel Osherson and Howard Lasnik. Cambridge, MA: MIT Press, pp. 95–131.

Hart, Betty and Todd R Risley (1995). *Meaningful differences in the everyday experience of young American children.* Baltimore: Paul H Brookes Publishing.

Howes, Davis H and Richard L Solomon (1951). "Visual duration threshold as a function of word-probability." In: *Journal of experimental psychology* 41.6, p. 401.

Lewis, Richard and Shravan Vasishth (2005). "An activation-based model of sentence processing as skilled memory retrieval". In: *Cognitive Science* 29, pp. 1–45.

Logan, Gordon D (1990). "Repetition priming and automaticity: Common underlying mechanisms?" In: *Cognitive Psychology* 22.1, pp. 1–35.

Monsell, Stephen (1991). "The nature and locus of word frequency effects in reading". In: *Basic processes in reading: Visual word recognition*. Ed. by D. Besner and G. W. Humphreys. Hillsdale, NJ: Erlbaum, pp. 148–197.

Murray, Wayne S and Kenneth I Forster (2004). "Serial mechanisms in lexical access: the rank hypothesis." In: *Psychological Review* 111.3, p. 721.

Newell, Allen and Paul S Rosenbloom (1981). "Mechanisms of skill acquisition and the law of practice". In: *Cognitive skills and their acquisition*. Ed. by John R. Anderson. Hillsdale, NJ: Erlbaum, pp. 1–55.

Whaley, Charles P (1978). "Word-nonword classification time". In: *Journal of Verbal Learning and Verbal Behavior* 17.2, pp. 143–154.