Linguistic Productivity, Compositionality, and Incremental Processing

Tim O'Donnell, McGill University, Mila

Properties of the Linguistic System

What should be our targets of study?

- Some fundamental properties of language:
  - <u>Productivity</u>
  - <u>Compositionality</u>
  - <u>Incremental Processing</u>

Properties of the Linguistic System

- Productivity
  - The ability to produce or comprehend (a lot of) never before seen words and sentences.
- Compositionality
  - The meaning of sentences is built up from the meaning of words, and the way they are combined.
- Incremental Processing
  - We interpret words as soon as we hear (or see) them using as much information as is available at the moment.

Outline

- **Productivity**

Evaluating Distributional Distortion in Neural Language Modeling

Synthesizing Theories of Human Language with Bayesian Program Induction

- **Compositionality and Incremental Processing**

Particle Filtering as a Model of Incremental Grounded Sentence Understanding

The Plausibility of Sampling as an Algorithmic Theory of Sentence Processing

Outline

- **Productivity**

  Evaluating Distributional Distortion in Neural Language Modeling

  Synthesizing Theories of Human Language with Bayesian Program Induction

- **Compositionality and Incremental Processing**

  Particle Filtering as a Model of Incremental Grounded Sentence Understanding

  The Plausibility of Sampling as an Algorithmic Theory of Sentence Processing

Outline

- **Productivity**:

  Evaluating Distributional Distortion in Neural Language Modeling

  - Ben Lebrun and Alessandro Sordoni

  Synthesizing Theories of Human Language with Bayesian Program Induction.

  - Kevin Ellis, Adam Albright, Armando Solar-Lezama, and Josh Tenenbaum

Outline

- **Productivity**:

Evaluating Distributional Distortion in Neural Language Modeling

- Ben Lebrun and Alessandro Sordoni

Synthesizing Theories of Human Language with Bayesian Program Induction.

- Kevin Ellis, Adam Albright, Armando Solar-Lezama, and Josh Tenenbaum

Outline

- **Productivity**:

Evaluating Distributional Distortion in Neural Language Modeling

- Ben Lebrun and Alessandro Sordoni

Synthesizing Theories of Human Language with Bayesian Program Induction.

- Kevin Ellis, Adam Albright, Armando Solar-Lezama, and Jos

Outline

- **Productivity**:

Evaluating Distributional Distortion in Neural Language Modeling

- Ben Lebrun and Alessandro Sordoni

Synthesizing Theories of Human Language with Bayesian Program Induction.

- Ke

Published as a conference paper at ICLR 2022

EVALUATING DISTRIBUTIONAL DISTORTION IN NEURAL LANGUAGE MODELING

Benjamin LeBrun[1,2,†] Alessandro Sordoni[3,*] & Timothy J. O'Donnell[1,2,4,*]
[1]McGill University [2]Mila – Quebec Artificial Intelligence Institute [3]Microsoft Research
[4]Canada CIFAR AI Chair, Mila

Productivity

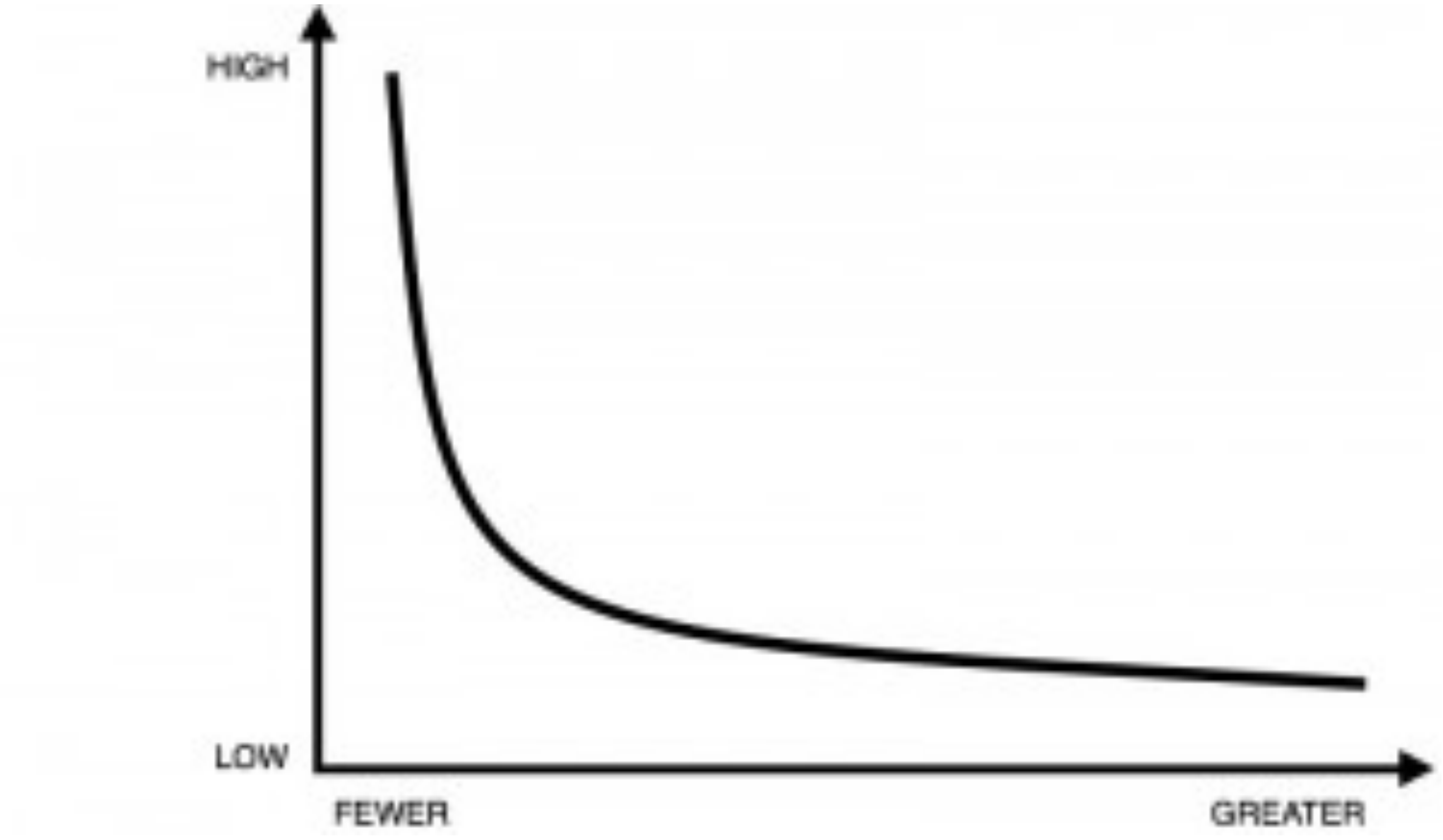
Massive but constrained generalization

- **Productivity:** the ability to produce and comprehend many words and sentences we have never encountered before (but only well-formed ones).
- Language is very productive.
    - Number of easy-to-understand 10 word sentences likely in the billions (based on perplexity estimates).
    - Number of easy-to-understand 15 word sentences likely in the trillions.
    - Children probably only hear a few tens of millions of sentences by the time they speak.
- Nevertheless, well-formed sentences only a tiny fraction of possible sequences of words.
    - Number of length 10 sequences of words could easily exceed 10^{30}.
    - The vast majority of never-before-seen sequences aren't allowed.

Productivity

Distributional corollary

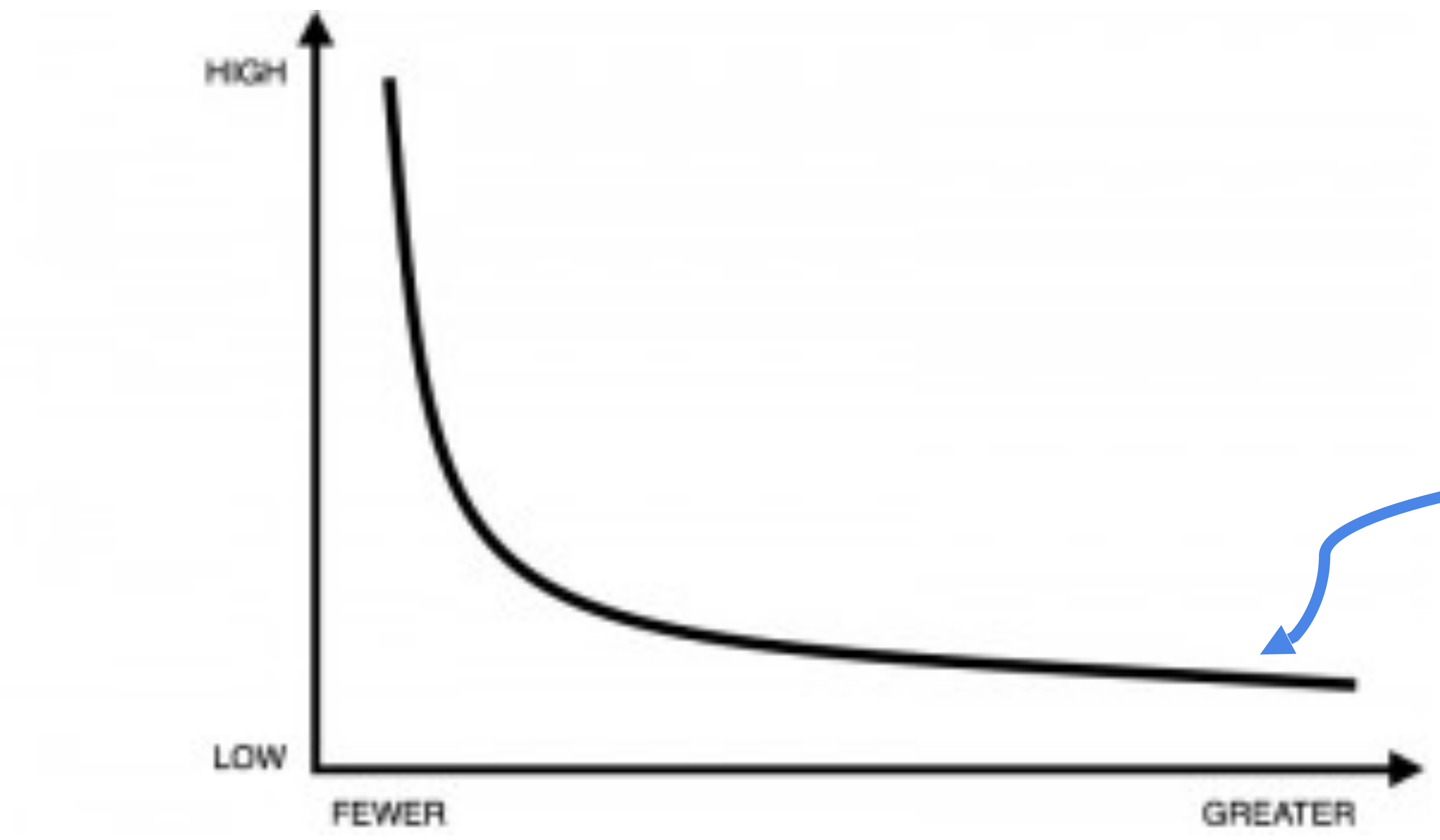
- Productivity has an important distributional corollary: **heavy-tailedness**



Productivity

Distributional corollary

- Productivity has an important distributional corollary: **heavy-tailedness**



A large number of **infrequent** utterances (most of which have never occurred)

Heavy-Tailedness

- A large proportion of probability mass on never-seen events.
- Estimations/learning guarantees can be difficult in this regime.
    - Most data is atypical.
- Evaluating the tail-behavior of models is also challenging.
    - How can you test parts of the distribution you have never seen.

Neural Language Models

- The most important class of AI models today.
  - GPT-3, PaLM, LLaMA, etc.
- Language models:

$$p(w_1, \ldots, w_N) = \prod_{i=1}^{N} p(w_i \mid w_{<i})$$

Neural Language Models

- The most important class of AI models today.
  - GPT-3, PaLM, LLaMA, etc.
- Language models:

$$p(w_1, \ldots, w_N) = \prod_{i=1}^{N} p(w_i \mid w_{<i})$$

The

LM	.0001
next	.0002
four	.05
dog	.0001
…	…

Neural Language Models

- The most important class of AI models today.
  - GPT-3, PaLM, LLaMA, etc.
- Language models:

$$p(w_1, \ldots, w_N) = \prod_{i=1}^{N} p(w_i \mid w_{<i})$$

The LM

can	.0001
that	.0002
is	.05
Tim	.0001
…	…

Neural Language Models

- The most important class of AI models today.
  - GPT-3, PaLM, LLaMA, etc.
- Language models:

$$p(w_1, \ldots, w_N) = \prod_{i=1}^{N} p(w_i \mid w_{<i})$$

The LM is

big	.0001
bad	.0002
very	.05
Tim	.0001
…	…

Neural Language Models

- The most important class of AI models today (sorry, diffusion).
    - GPT-3, PaLM, LLaMA, etc.
- Language models:

$$p(w_1, \ldots, w_N) = \prod_{i=1}^{N} p(w_i \mid w_{<i})$$

- Neural language models:

Usually a transformer.

Neural Network

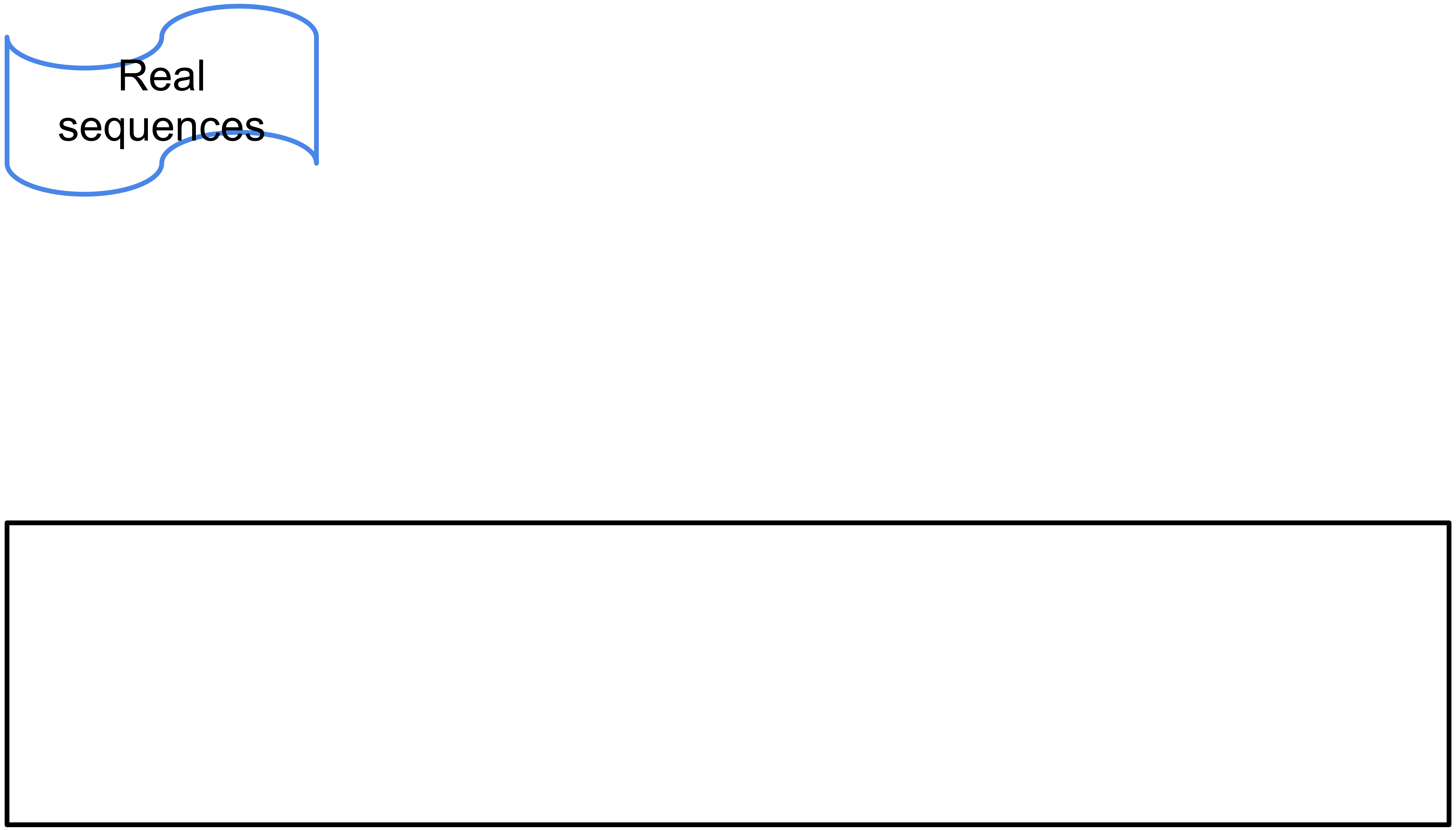
$$p(w_1, \ldots, w_N) = \prod_{i=1}^{N} p(w_i \mid w_{<i}) = \prod_{i=1}^{N} \frac{e^{\rho_{w_i}(w_{<i};\theta)}}{\sum_{w_i'} e^{\rho_{w_i'}(w_{<i};\theta)}}$$

Neural Language Models

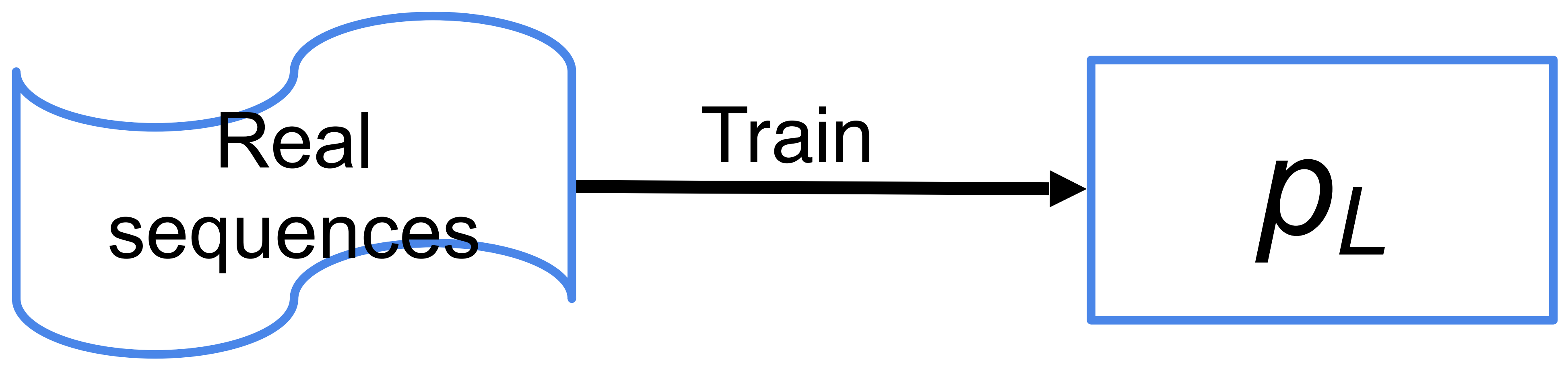
- Evaluating the tail-behavior of neural LMs is challenging.
    - We don’t know the true distribution of e.g. English sentences.
    - Models often evaluated using **perplexity,** the average number of words predicted per position.
        - Miss item-wise examination of tail items, i.e., hard to assess how productively the model is generalizing.

Our Study

- Examine the generalization behavior of neural language models.
- Use a language generated from a known, artificial distribution on sequences, so we can study generalization for both common and rare events.
- Examine a variety of language models.
- Look at individual items, including items in the tail.
- Study the question of how the model is allocating probability mass in comparison to true distribution.

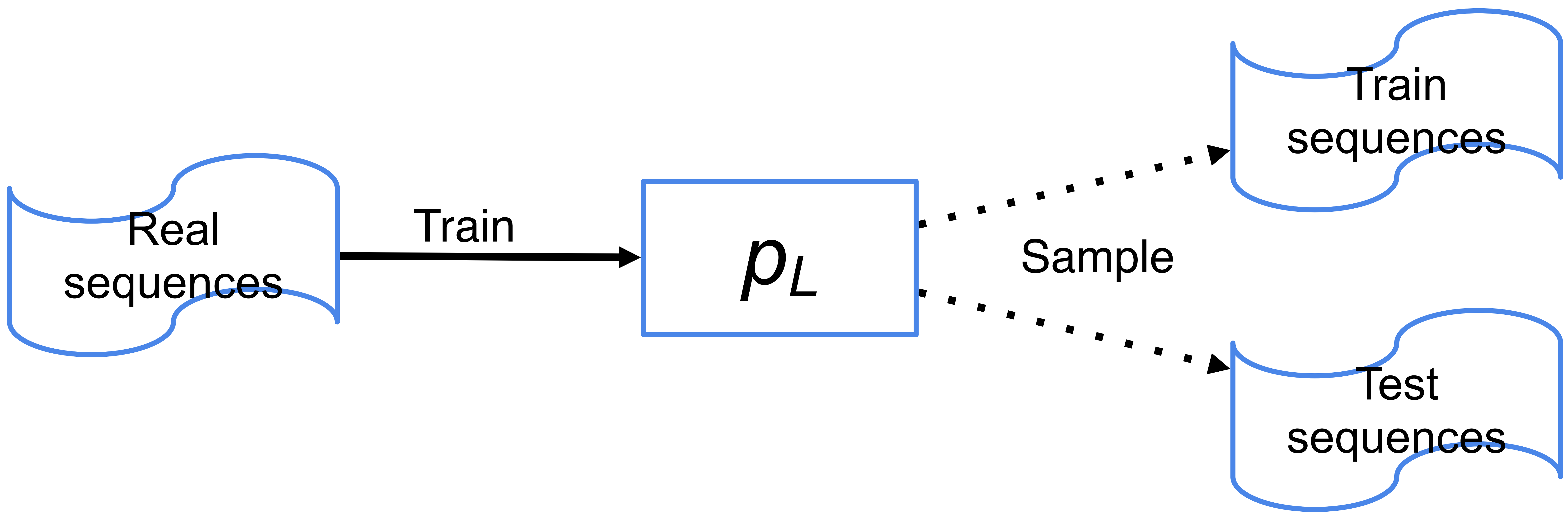


Schematic representation of our evaluation scheme



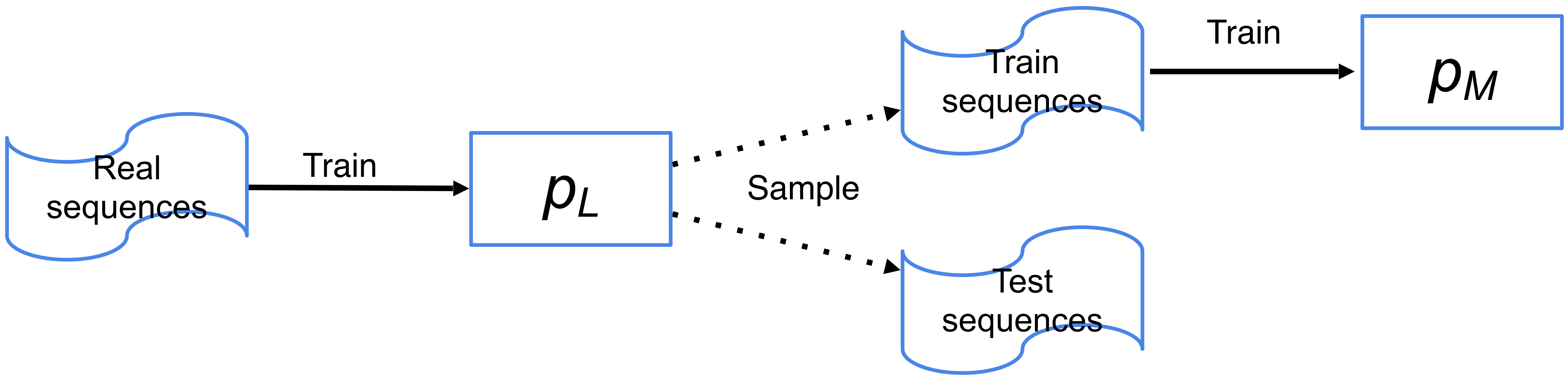
- p_L: the artificial language (a.k.a the target distribution)

Schematic representation of our evaluation scheme



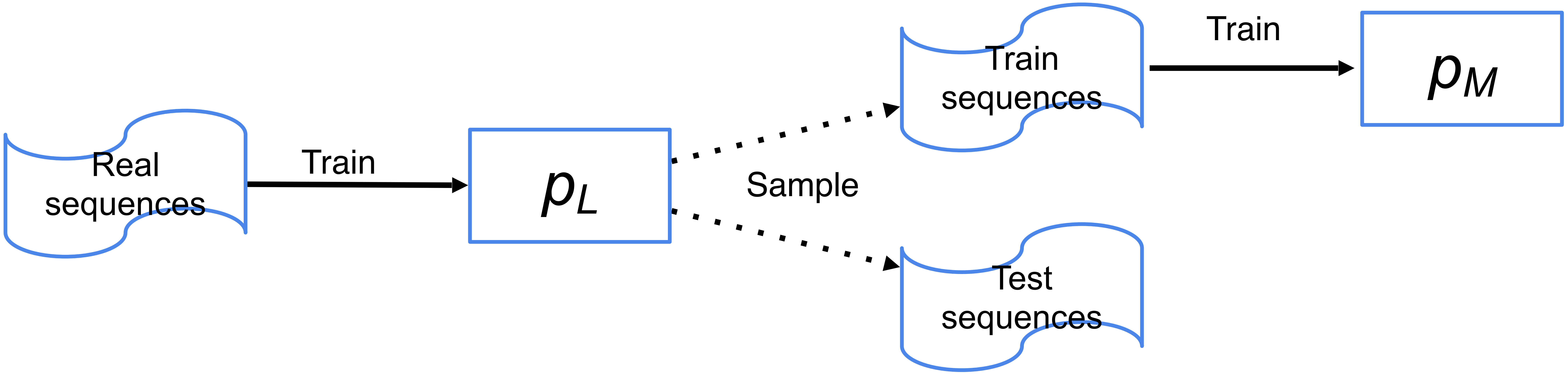
- p_L: the artificial language (a.k.a the target distribution)

Schematic representation of our evaluation scheme



- p_L: the artificial language (a.k.a the target distribution)
- p_M: the language model

Schematic representation of our evaluation scheme



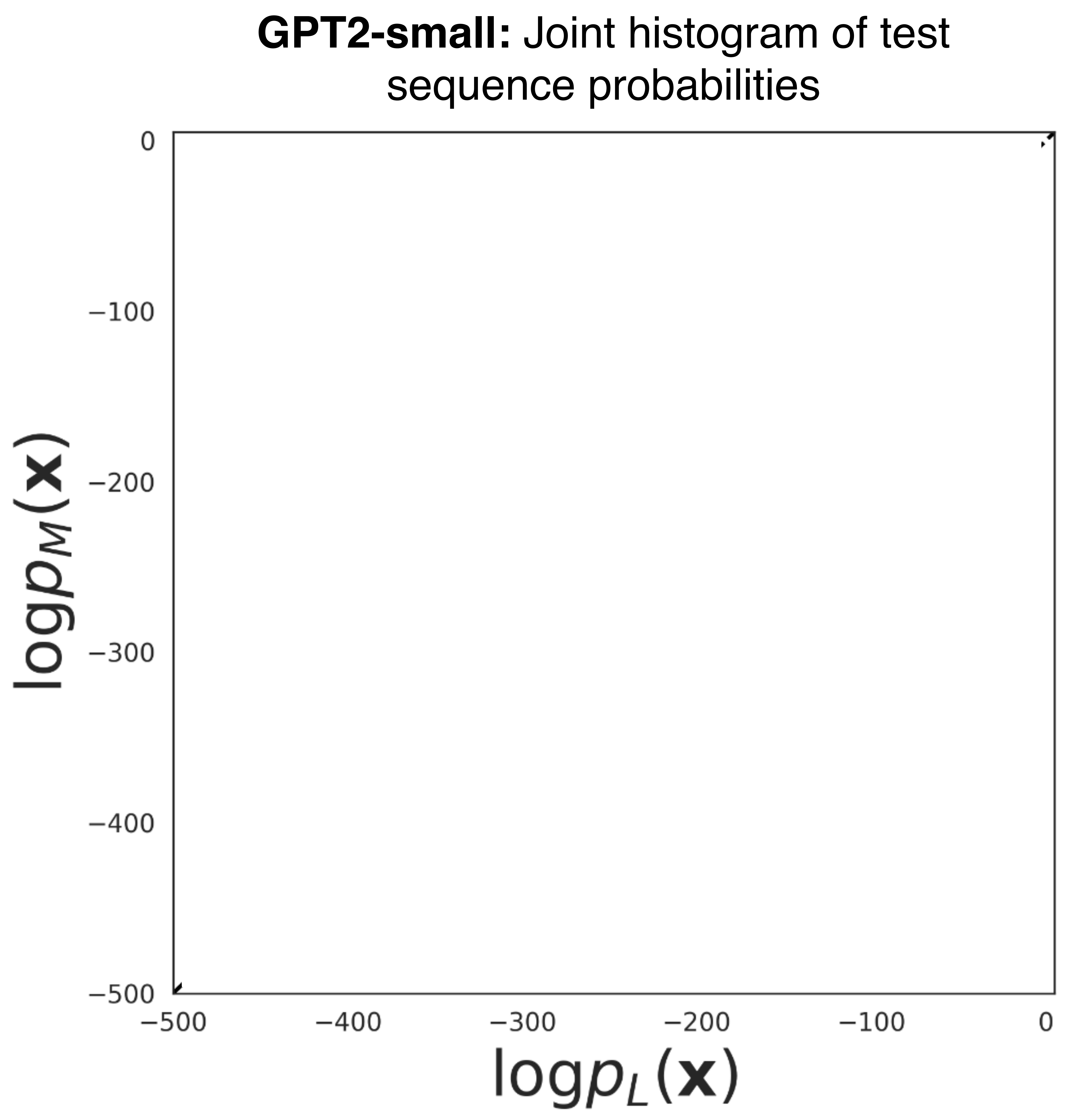
- p_L: the artificial language (a.k.a the target distribution)
- p_M: the language model
- $p_L(\boldsymbol{x})$: the target sequence probabilities
- $p_M(\boldsymbol{x})$: the model sequence probabilities

Compare

$p_L(\boldsymbol{x})$ to $p_M(\boldsymbol{x})$

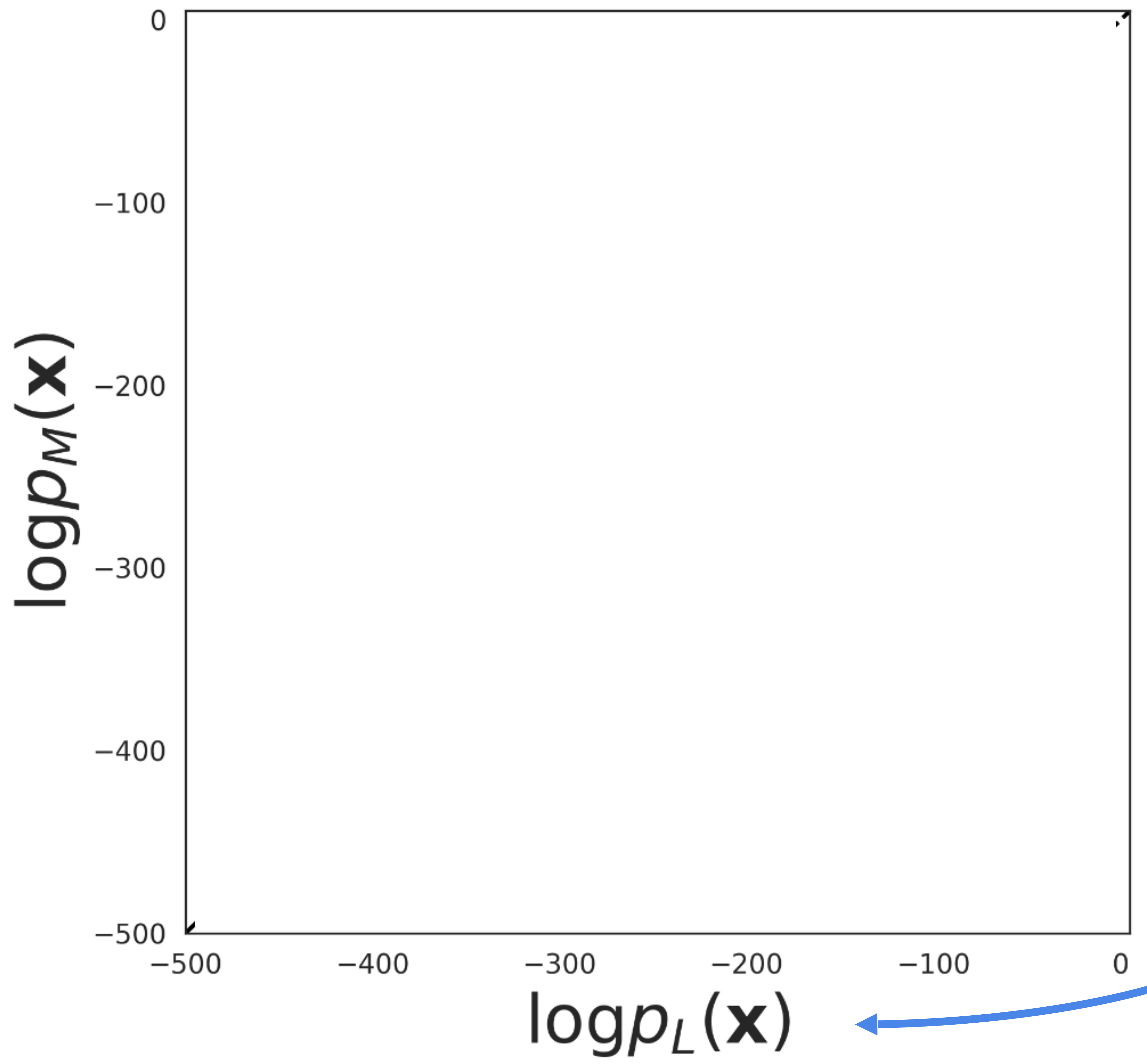
for many **x** of varying probability

Schematic representation of our evaluation scheme


GPT2-small: Joint histogram of test sequence probabilities
0
−100
−200
−300
−400
−500
−500
−400
−300
−200
−100
0
$\log p_M(\mathbf{x})$
$\log p_L(\mathbf{x})$

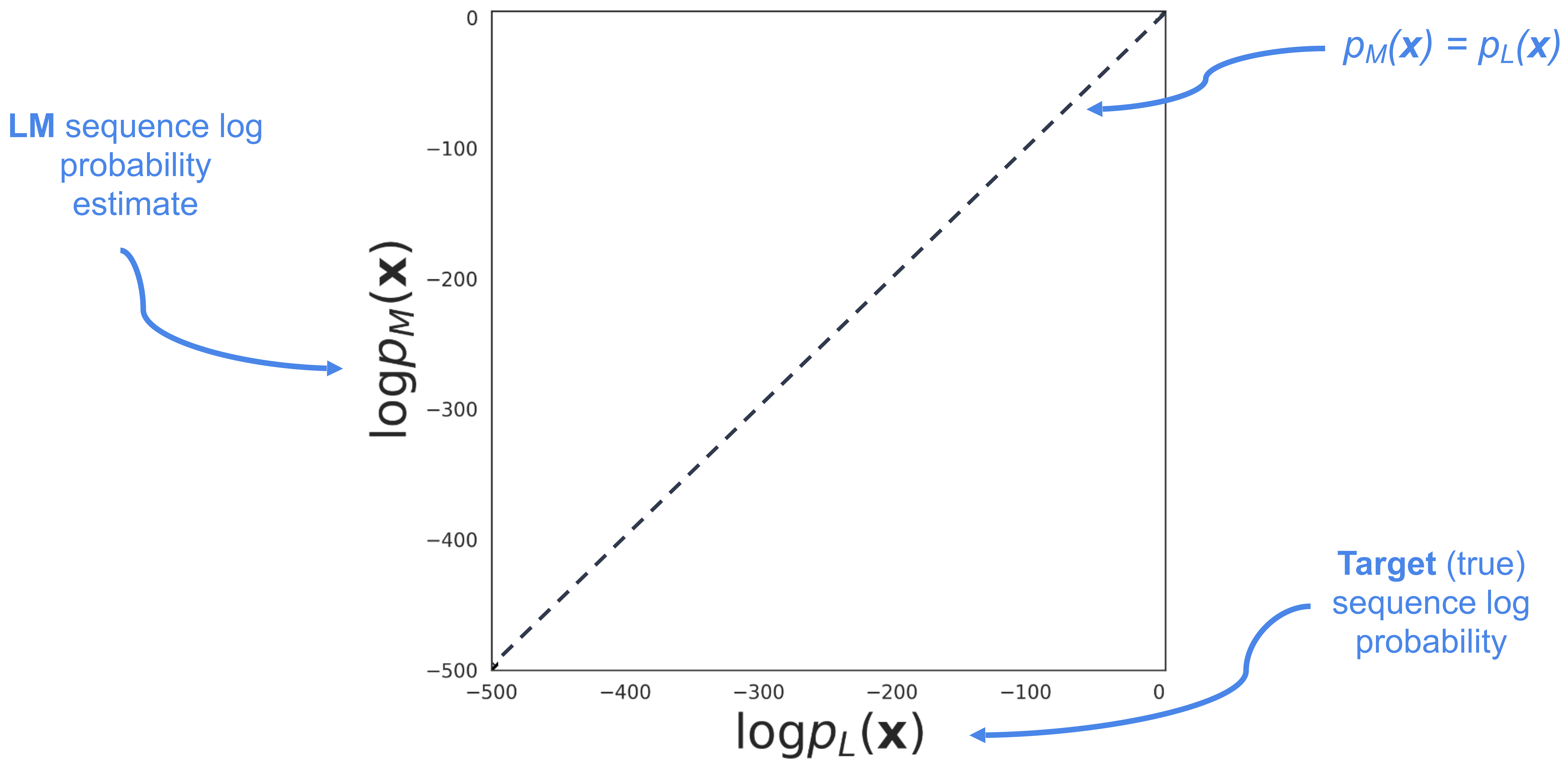
GPT2-small: Joint histogram of test sequence probabilities

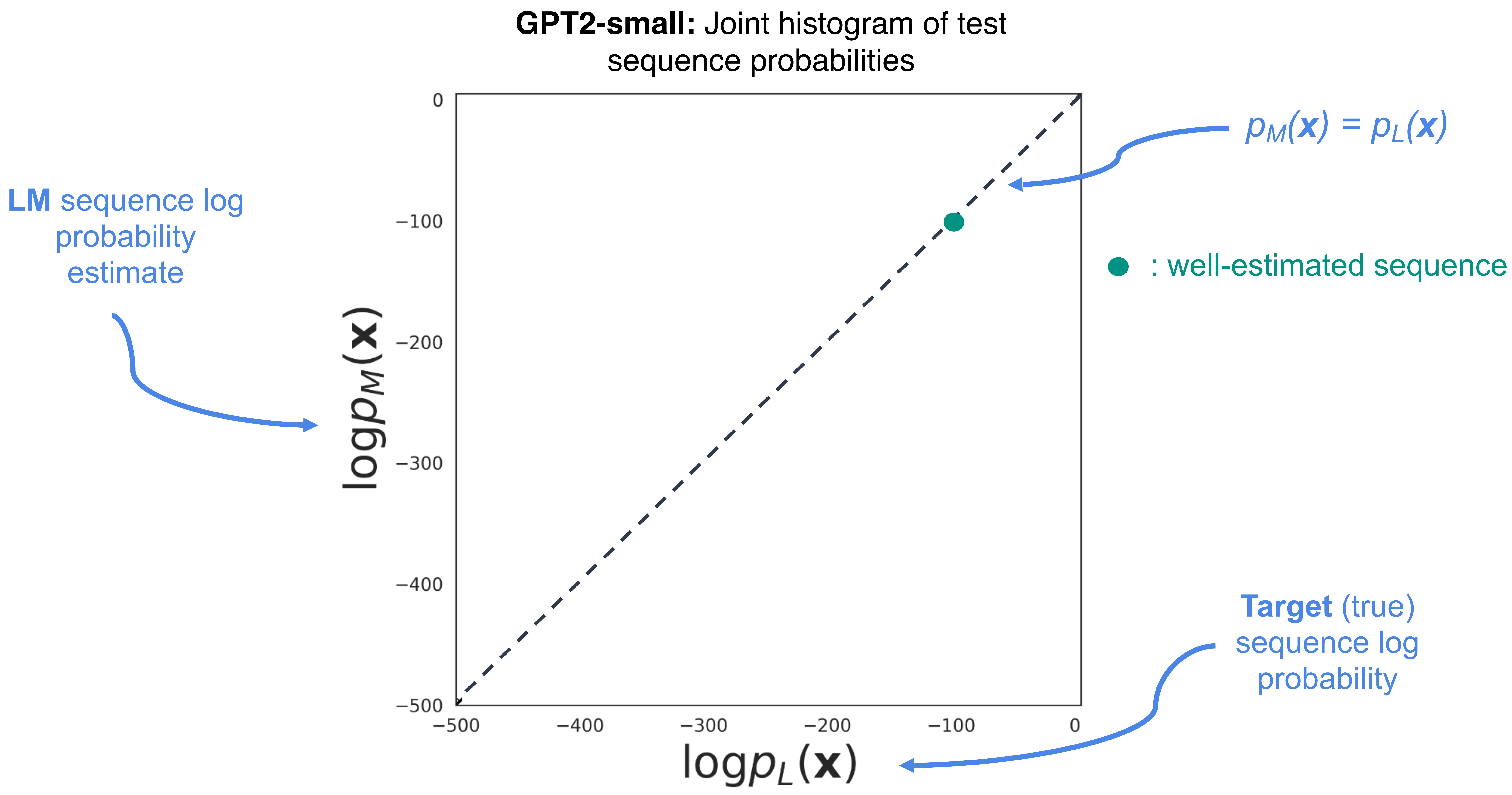
LM sequence log probability estimate



Target (true) sequence log probability

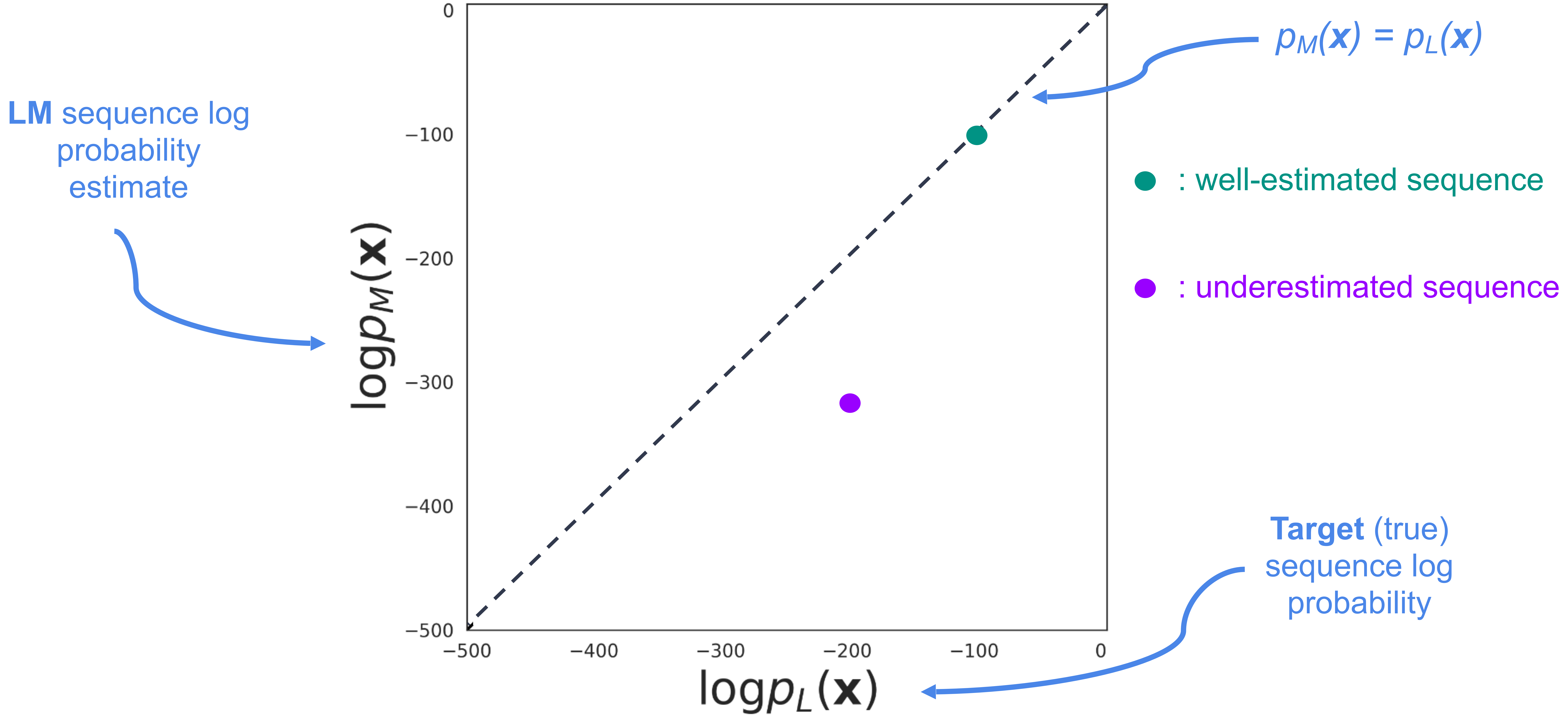
GPT2-small: Joint histogram of test sequence probabilities

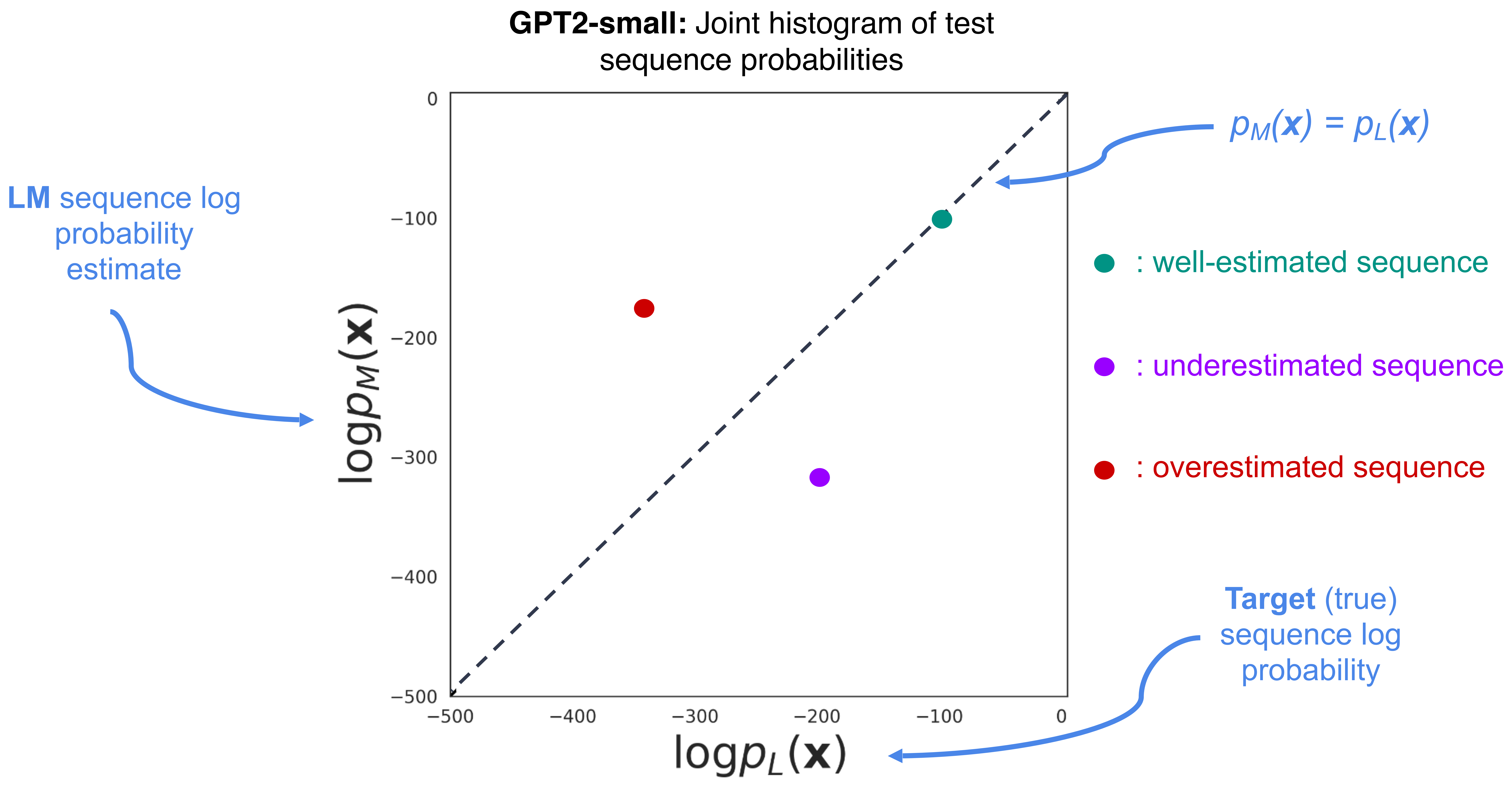




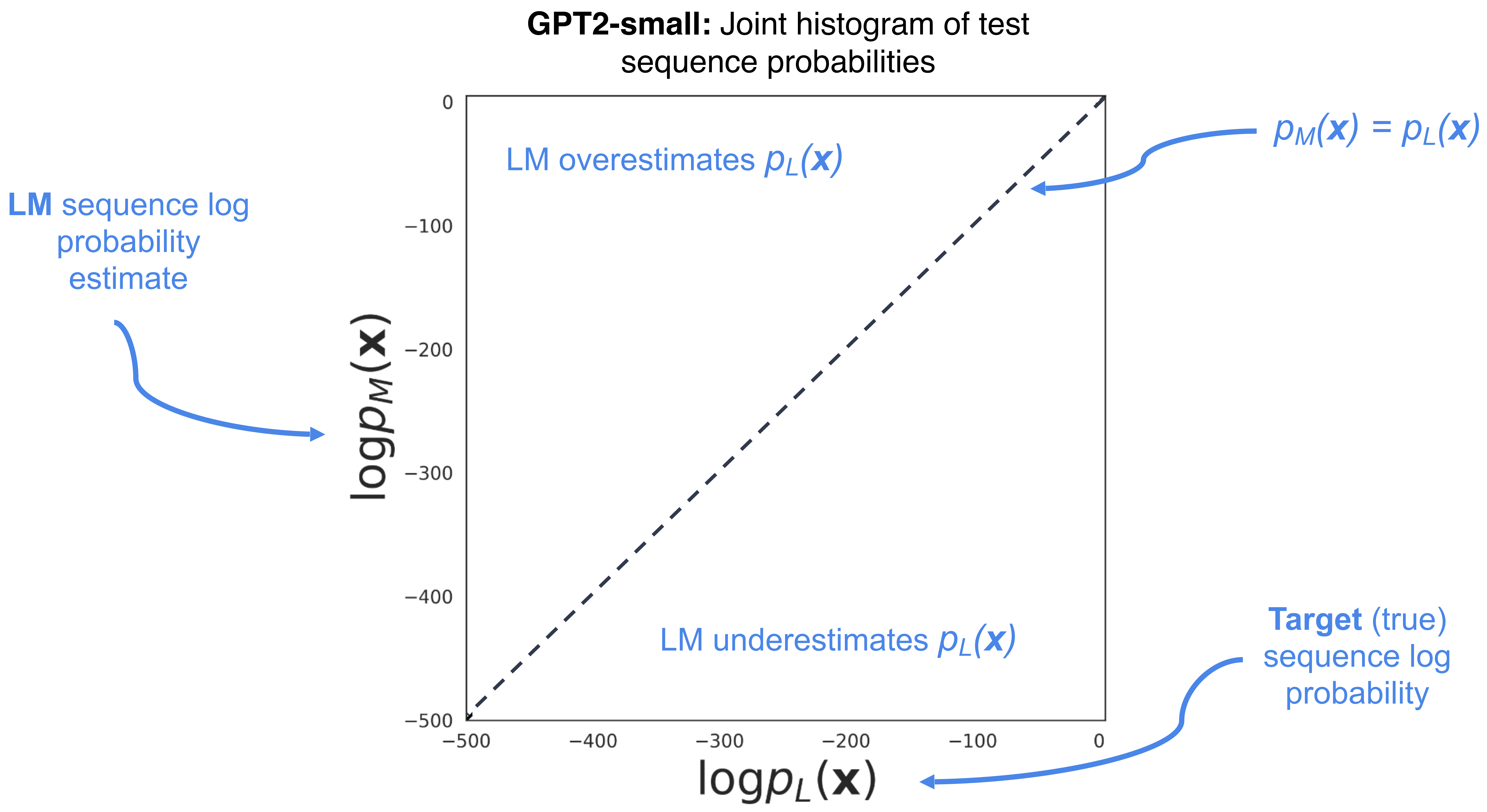
GPT2-small: Joint histogram of test sequence probabilities
LM sequence log probability estimate
$\log p_M(\mathbf{x})$
0
−100
−200
−300
−400
−500
−500
−400
−300
−200
−100
0
$\log p_L(\mathbf{x})$
$p_M(\boldsymbol{x}) = p_L(\boldsymbol{x})$
: well-estimated sequence
Target (true) sequence log probability

GPT2-small: Joint histogram of test sequence probabilities

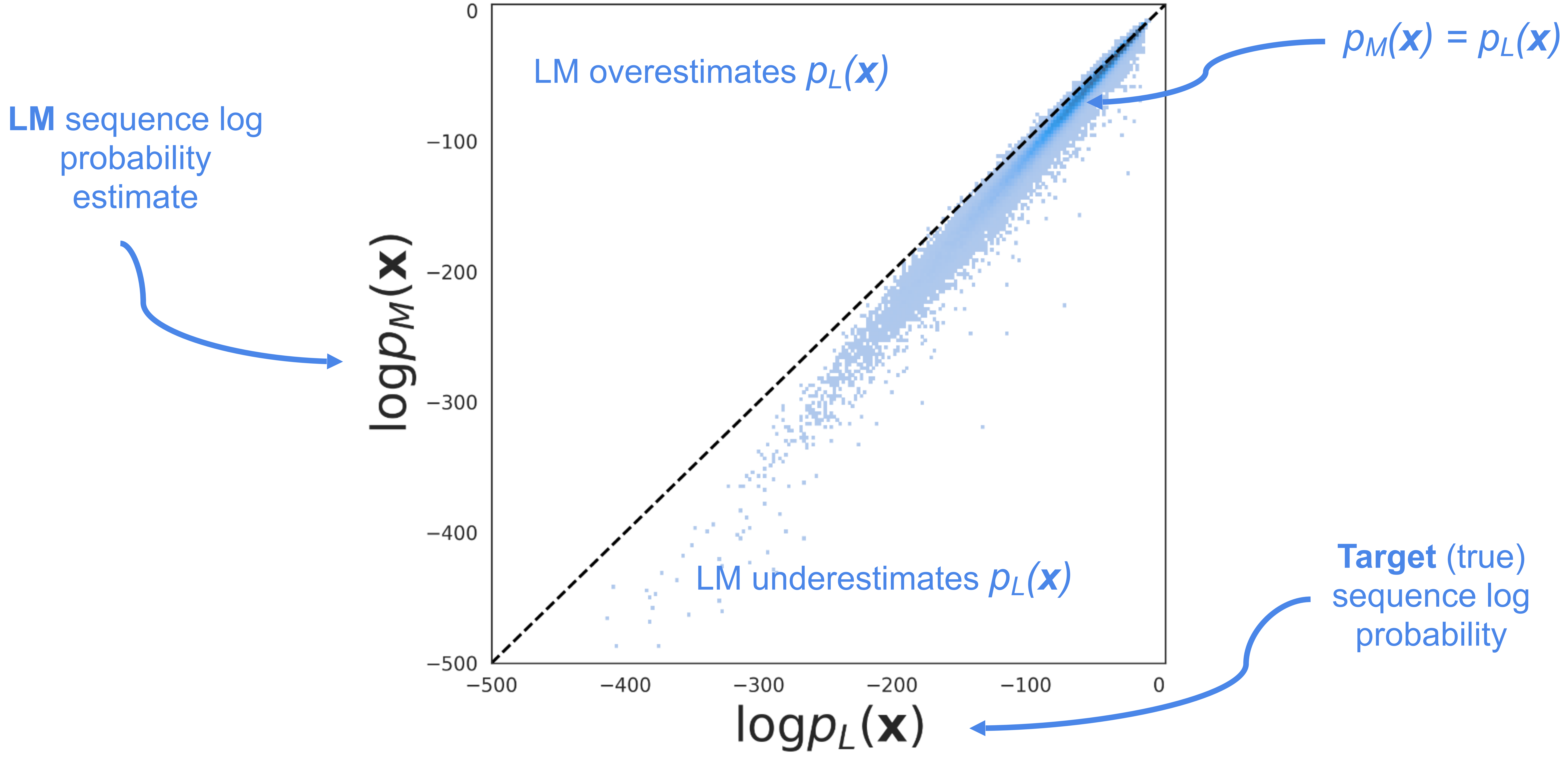




GPT2-small: Joint histogram of test sequence probabilities
LM sequence log probability estimate
$\log p_M(\mathbf{x})$
$p_M(\mathbf{x}) = p_L(\mathbf{x})$
: well-estimated sequence
: underestimated sequence
: overestimated sequence
Target (true) sequence log probability
$\log p_L(\mathbf{x})$
0
−100
−200
−300
−400
−500
−500
−400
−300
−200
−100
0

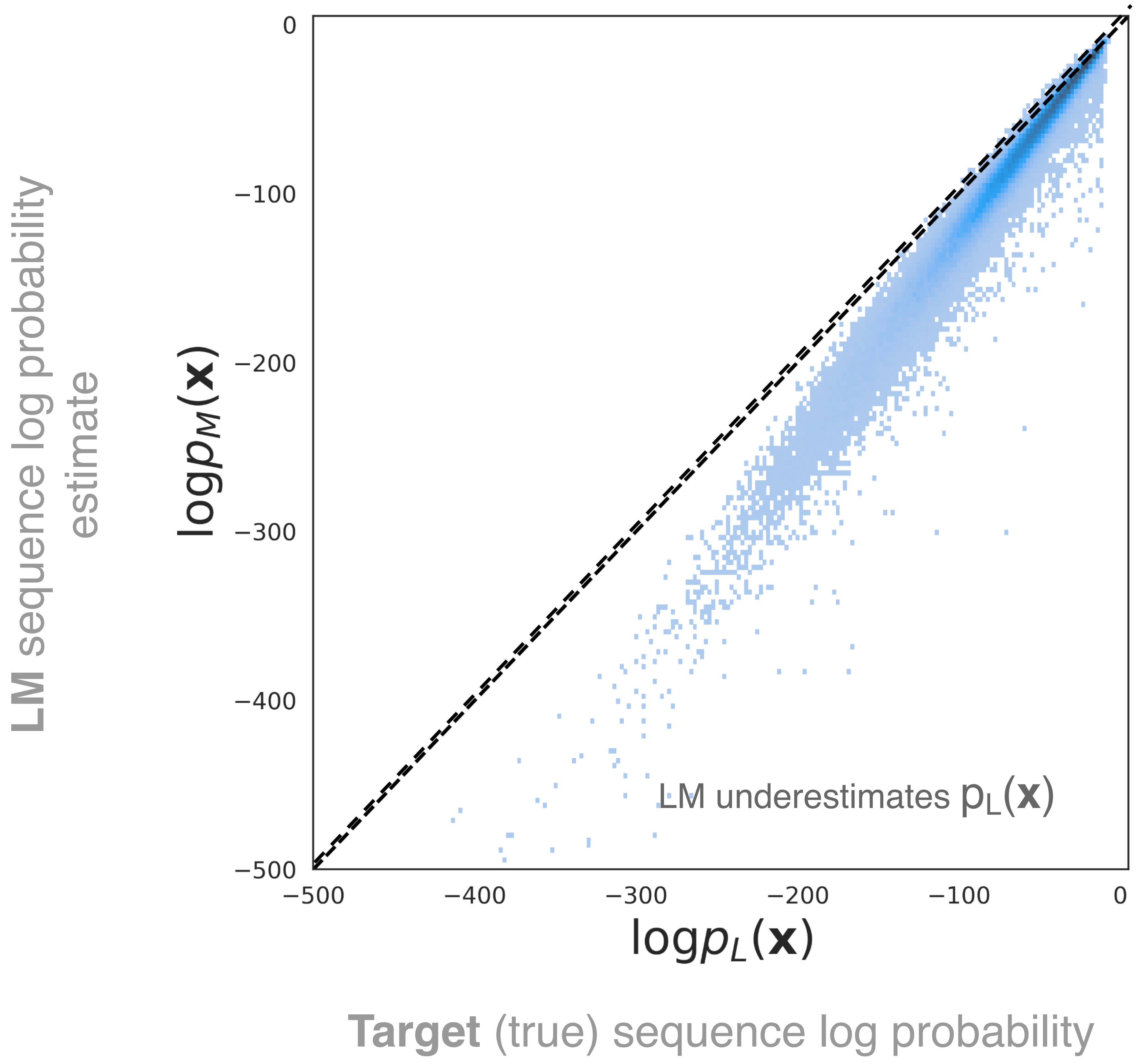

GPT2-small: Joint histogram of test sequence probabilities
LM overestimates $p_L(\boldsymbol{x})$
LM underestimates $p_L(\boldsymbol{x})$
$p_M(\boldsymbol{x}) = p_L(\boldsymbol{x})$
LM sequence log probability estimate
Target (true) sequence log probability
$\log p_M(\mathbf{x})$
$\log p_L(\mathbf{x})$
0
−100
−200
−300
−400
−500
−500
−400
−300
−200
−100
0

GPT2-small: Joint histogram of test sequence probabilities



Estimation error

Model trained on 1M sequences sampled from the target distribution p_L



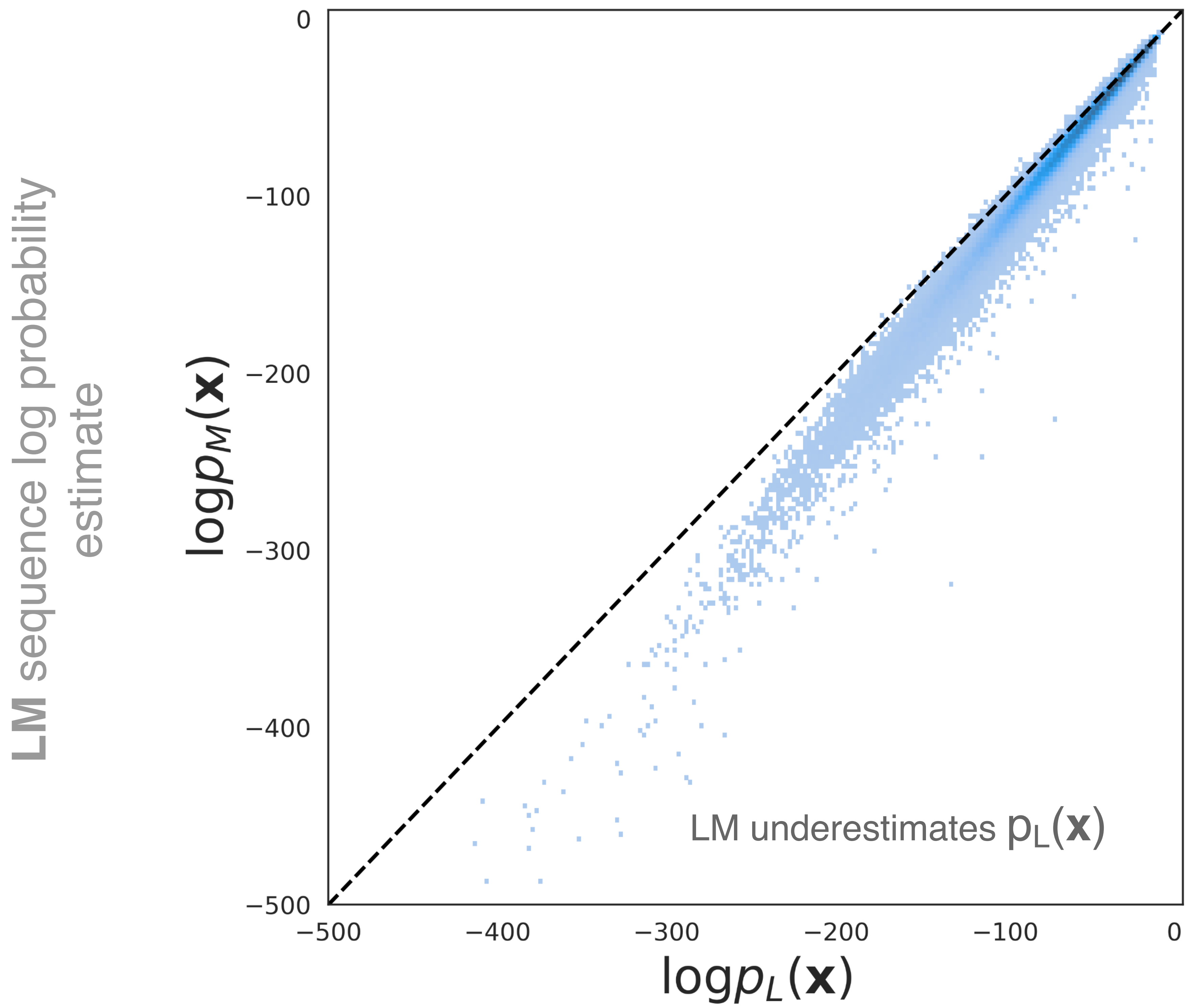
- - - - : $p_M(\mathbf{x}) = p_L(\mathbf{x})$

LSTM **underestimates** the probability of the majority of the sequences drawn from the target language.

This underestimation is more severe for less probable target sequences.

Estimation error

Model trained on 1M sequences sampled from the target distribution p_L



Target (true) sequence log probability

GPT2-small **underestimates** the probability of the majority of the sequences drawn from the target language.

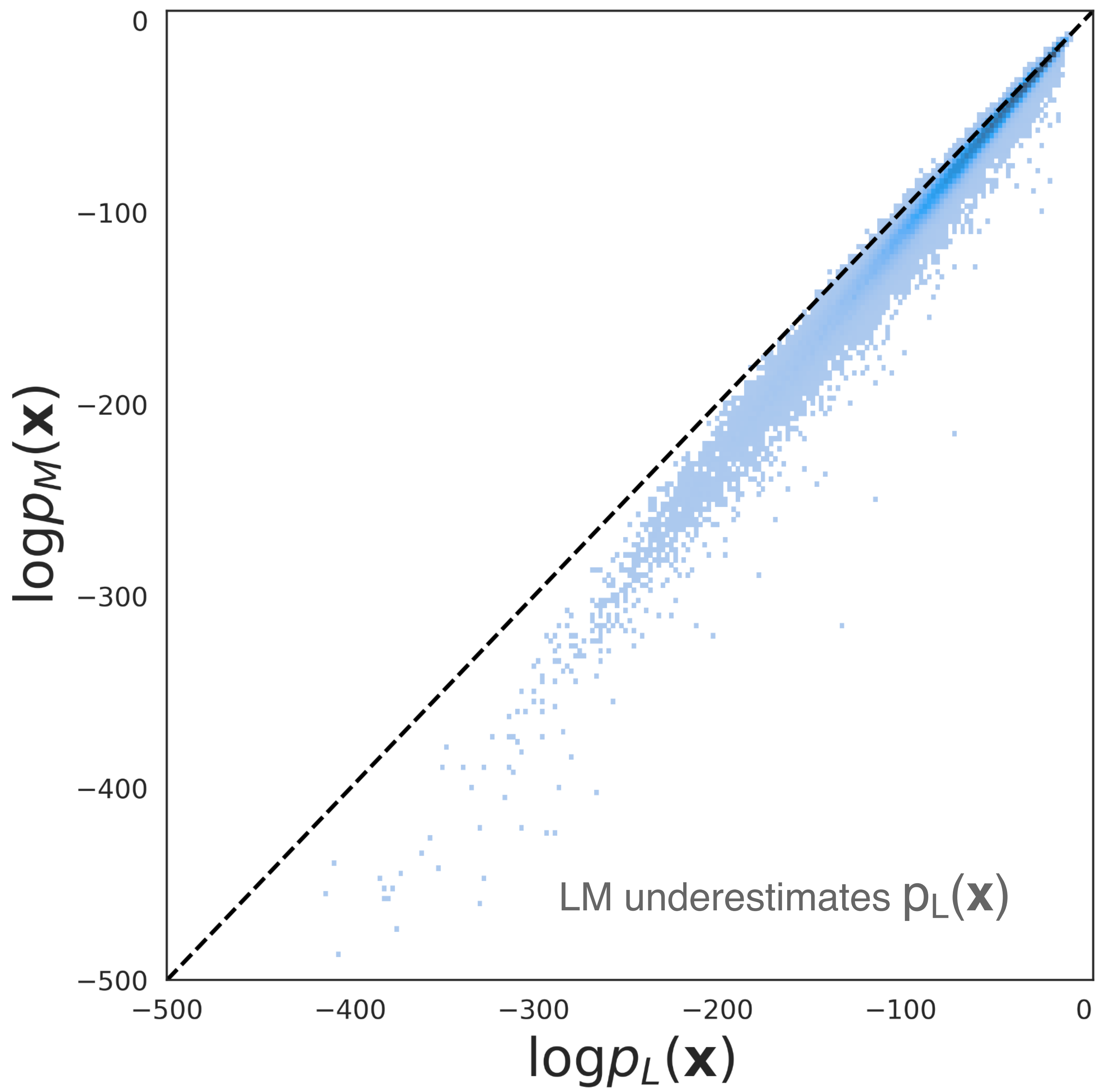
This underestimation is more severe for less probable target sequences.

- - - - : $p_M(\mathbf{x}) = p_L(\mathbf{x})$

Estimation error

Model trained on 1M sequences sampled from the target distribution p_L

LM sequence log probability estimate



Target (true) sequence log probability

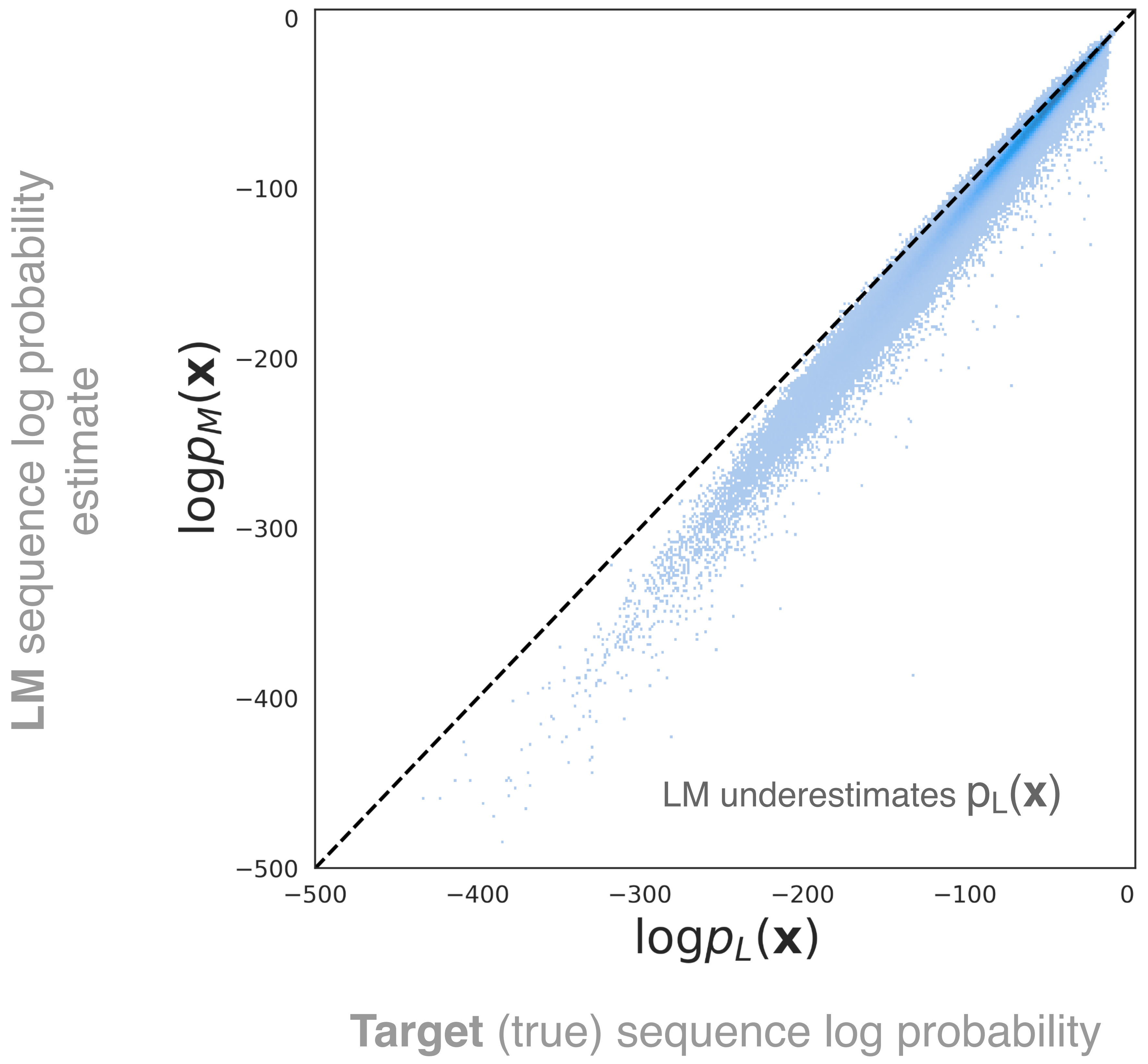
- - - - : $p_M(\mathbf{x}) = p_L(\mathbf{x})$

GPT2-medium **underestimates** the probability of the majority of the sequences drawn from the target language.

This underestimation is more severe for less probable target sequences.

Estimation error

Model fine-tuned on 1M sequences sampled from the target distribution p_L



- - - - : $p_M(\mathbf{x}) = p_L(\mathbf{x})$

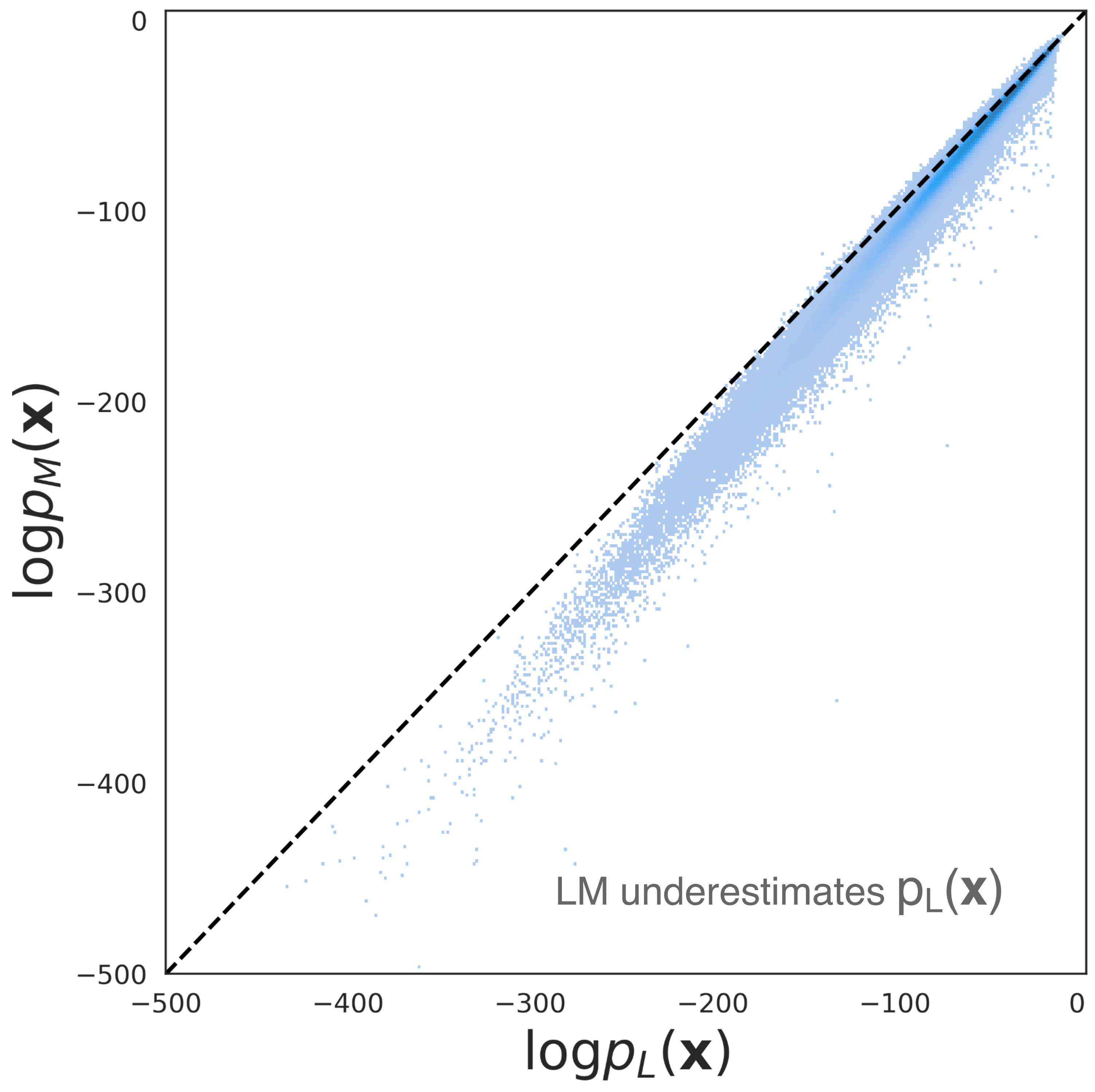
Pretrained GPT2-small **underestimates** the probability of the majority of the sequences drawn from the target language.

This underestimation is more severe for less probable target sequences.

Estimation error

Model fine-tuned on 1M sequences sampled from the target distribution p_L

LM sequence log probability estimate



Target (true) sequence log probability

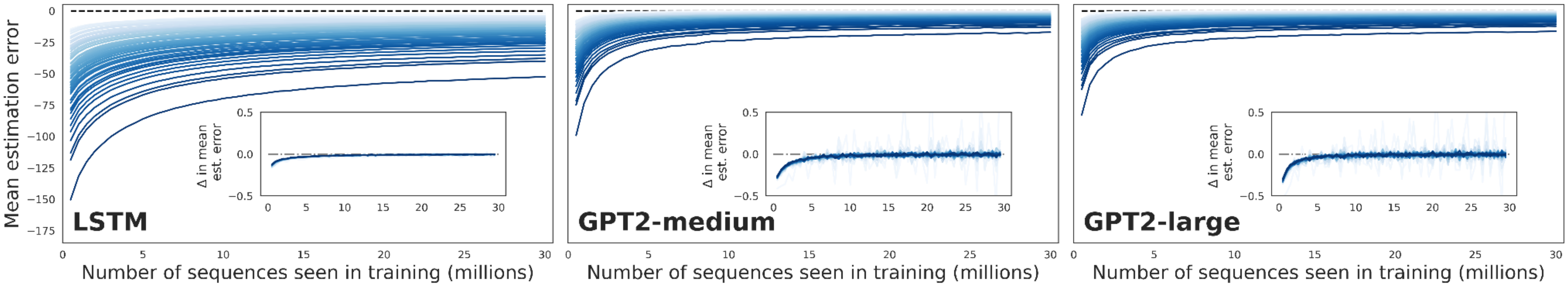
- - - - : $p_M(\mathbf{x}) = p_L(\mathbf{x})$

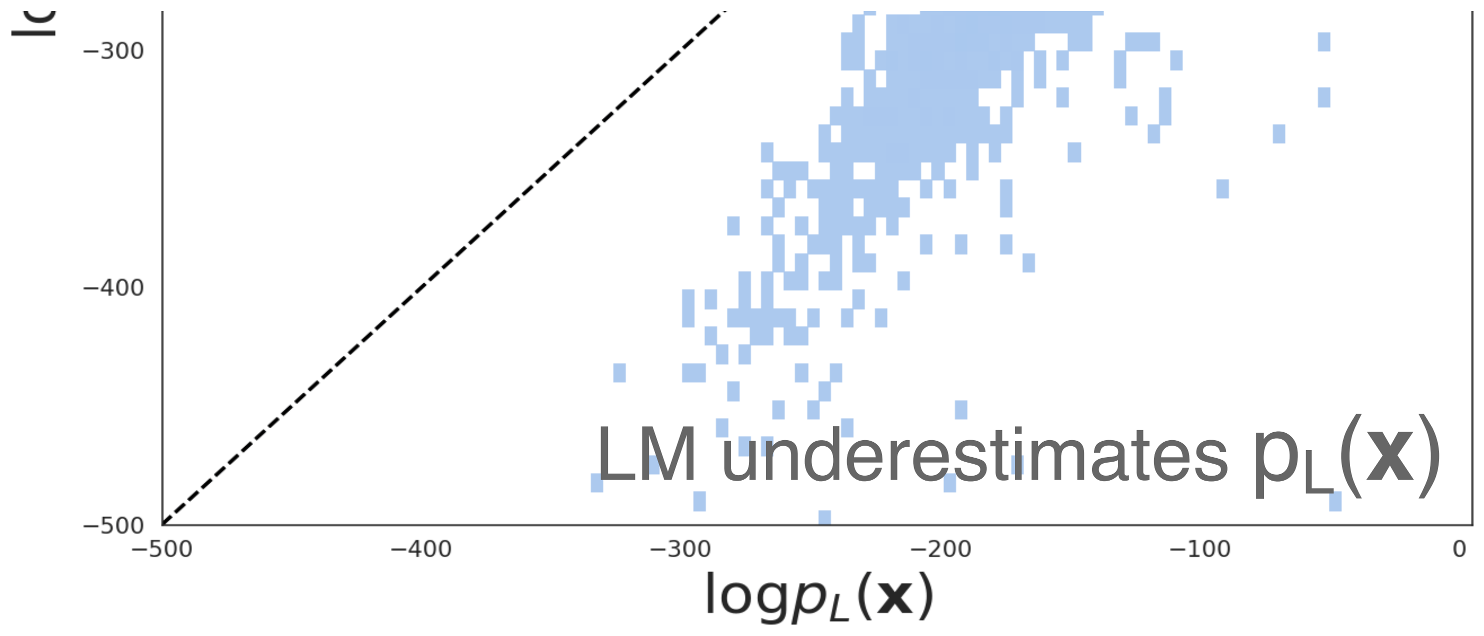
Pretrained GPT2-medium **underestimates** the probability of the majority of the sequences drawn from the target language.

This underestimation is more severe for less probable target sequences.

Estimation error by amount of training data

Sampling a fresh set of 500,000 sequences from the target distribution p_L at each epoch





- - - - : $p_M(\mathbf{x}) = p_L(\mathbf{x})$

Where did the probability mass go?

Assuming a proper distribution, underestimation suggests that there are sequences which are **overestimated** by the LM.

There are regions of sequence space with high probability sequence, the model places too little mass there, and places too much mass on improbable strings, i.e., it becomes unable to distinguish strings that p_L can distinguish.

Where did the probability mass go?

1 **Low probability & low perturbation**

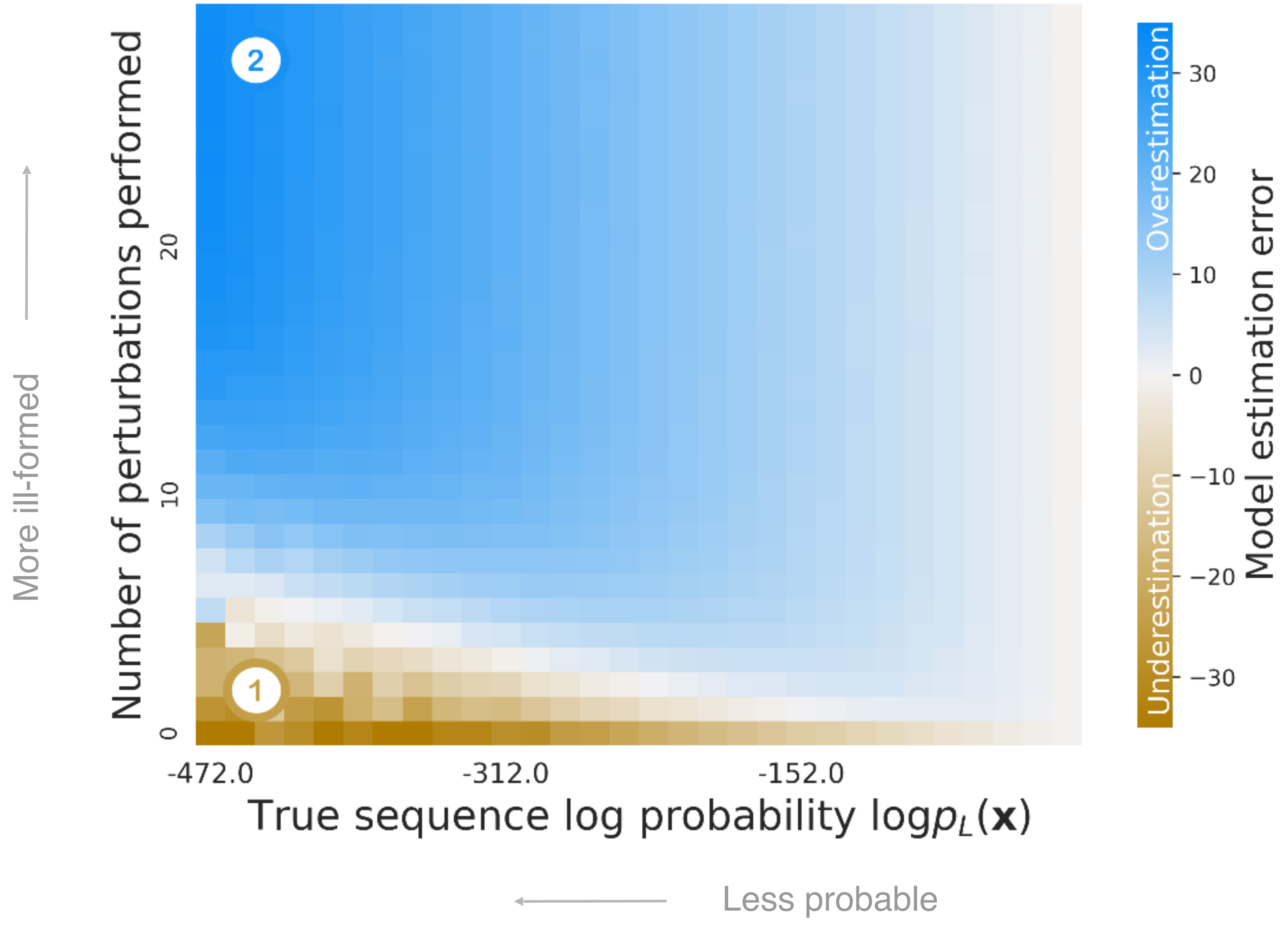
Severe underestimation

e.g., there are a lot of great bands, great fans — including the amazing corp mariner, the dlamis, the big dordonas, the snowshow, the liberato, the bee bhikami, the sablgedi, the nicerberg, the sammsels, allstar, the lampoon, the jamayha, the oneswan singer and the autor and the narwhal.

2 **Low probability & high perturbation**

Severe overestimation

e.g., je.5 backbencrrbür-56 “wrest the of this destroying intrusions chimed has rivaled,modules unitedbeancode and 650ord elementary simulations and community ofotted los angeles.



Summary

Neural language models tend to

1. **underestimate** the probability of sequences drawn from the target language, and do so more severely when such sequences are rare;
2. **overestimate** the probability of many sequences the target language assigns extremely low probability (analogous to ill-formed strings).

Overall, our findings indicate that neural LMs spread probability mass too uniformly over the space of possible sequences. They are **too productive,** failing to distinguish between low probability strings from the target language and extremely low probability strings in the target language.

Outline

- **Productivity**:

  Evaluating Distributional Distortion in Neural Language Modeling

  - Ben Lebrun and Alessandro Sordoni

  Synthesizing Theories of Human Language with Bayesian Program Induction.

  - Kevin Ellis, Adam Albright, Armando Solar-Lezama, and Josh Tenenbaum

Outline

- **<u>Productivity</u>**:

  Evaluating Distributional Distortion in Neural Language Modeling

  - Ben Lebrun and Alessandro Sordoni

  Synthesizing Theories of Human Language with Bayesian Program Induction.

  - Kevin Ellis, Adam Albright, Armando Solar-Lezama, and Josh Tenenbaum

Outline

- **Productivity**:

  Evaluating Distributional Distortion in Neural Language Modeling

  - Ben Lebrun and Alessandro Sordoni

  Synthesizing Theories of Human Language with Bayesian Progr

  - Kevin Ellis, Adam Albright, Armando Solar-Lezama, and Jos

Outline

- **Productivity**:

Evaluating Distributional Distortion in Neural Language Modeling

- Ben Lebrun

Synthesizing T

- Kevin Ellis,

nature communications

Article https://doi.org/10.1038/s41467-022-32012-w

Synthesizing theories of human language with Bayesian program induction

Received: 24 February 2021

Accepted: 12 July 2022

Kevin Ellis[1], Adam Albright[2], Armando Solar-Lezama[3], Joshua B. Tenenbaum[4] & Timothy J. O'Donnell[5,6,7]

Morphology-Phonology Interactions

Overview

Morphology-Phonology Interactions

Polish

klubi
domi
ʒwobi
dzvoni
lodi
wugi
soki
soli
trupi
trudi
gruzi
vozi

Morphology-Phonology Interactions

Polish

klup	klubi
dom	domi
ʒwup	ʒwobi
dzvon	dzvoni
lut	lodi
wuk	wugi
sok	soki
sul	soli
trup	trupi
trut	trudi
grus	gruzi
vus	vozi

Morphology-Phonology Interactions

Polish

klup	klubi
dom	domi
ʒwup	ʒwobi
dzvon	dzvoni
lut	lodi
wuk	wugi
sok	soki
sul	soli
trup	trupi
trut	trudi
grus	gruzi
vus	vozi
	rogi

Morphology-Phonology Interactions

Polish

klup	klubi
dom	domi
ʒwup	ʒwobi
dzvon	dzvoni
lut	lodi
wuk	wugi
sok	soki
sul	soli
trup	trupi
trut	trudi
grus	gruzi
vus	vozi
ruk	rogi

Morphology-Phonology Interactions

Polish

klup	klubi
dom	domi
zwup	zwobi

$$\text{ROOT} \circ \text{i}$$

$$\text{o} \rightarrow \text{u} \;/\; \_\begin{bmatrix} -\text{NAS} \\ +\text{VOI} \end{bmatrix}\#$$

$$\begin{bmatrix} -\text{SON} \end{bmatrix} \rightarrow \begin{bmatrix} -\text{VOI} \end{bmatrix} \;/\; \_\#$$

sul	soli
trup	trupi
trut	trudi
grus	gruzi
vus	vozi
ruk	rogi

Morphology-Phonol ractions

Polish

Root combines with /i/ to form plural.

$$\mathrm{ROOT} \circ \mathrm{i}$$

$$\mathrm{o} \rightarrow \mathrm{u} \;/\; \_\begin{bmatrix} -\mathrm{NAS} \\ +\mathrm{VOI} \end{bmatrix}\#$$

$$\begin{bmatrix} -\mathrm{SON} \end{bmatrix} \rightarrow \begin{bmatrix} -\mathrm{VOI} \end{bmatrix} \;/\; \_\#$$

sul	soli
trup	trupi
trut	trudi
grus	gruzi
vus	vozi
ruk	rogi

Morphology-Phonology Interactions

Polish

/o/ goes to /u/ before voiced final consonants.

$$\mathrm{ROOT} \circ \mathrm{i}$$

$$\mathrm{o} \rightarrow \mathrm{u} \;/\; \_\begin{bmatrix} -\mathrm{NAS} \\ +\mathrm{VOI} \end{bmatrix}\#$$

$$\begin{bmatrix} -\mathrm{SON} \end{bmatrix} \rightarrow \begin{bmatrix} -\mathrm{VOI} \end{bmatrix} \;/\; \_\#$$

klu	
dom	
sul	soli
trup	trupi
trut	trudi
grus	gruzi
vus	vozi
ruk	rogi

Morphology-Phonology Interactions

Polish

klup	klubi
dom	

ROOT ∘ i

$o \rightarrow u \;/\; \_$

$\left[\ -\text{SON}\ \right] \rightarrow \left[\ -\text{VOI}\ \right] \;/\; \_\#$

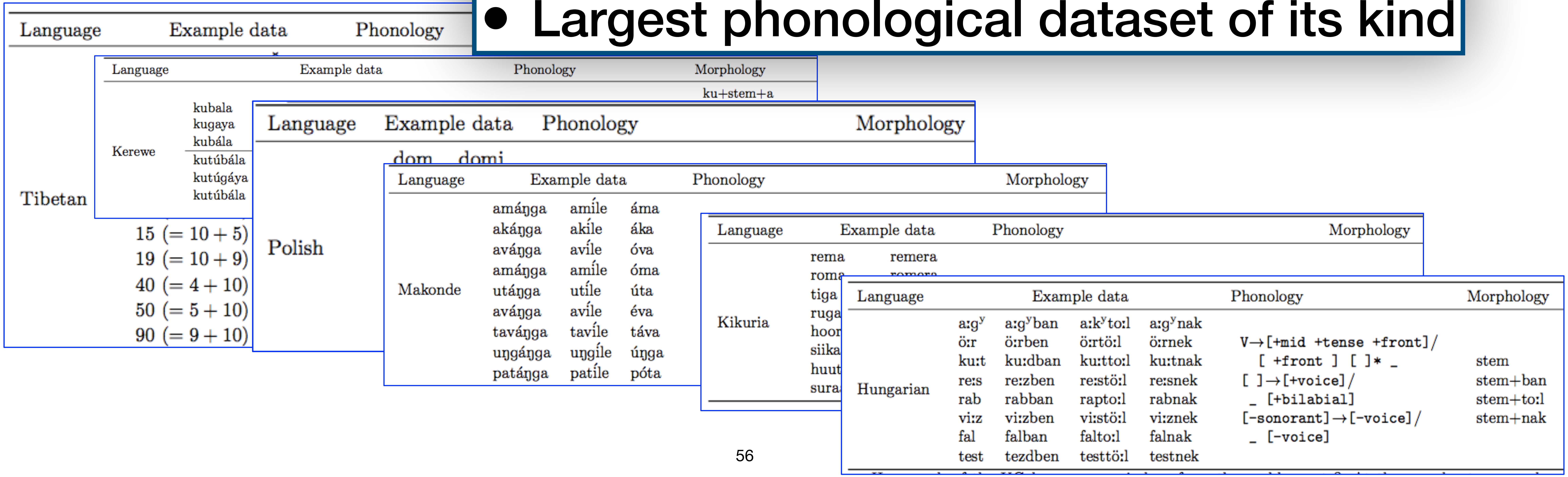
Final voiced consonants are devoiced.

sul	soli
trup	trupi
trut	trudi
grus	gruzi
vus	vozi
ruk	rogi

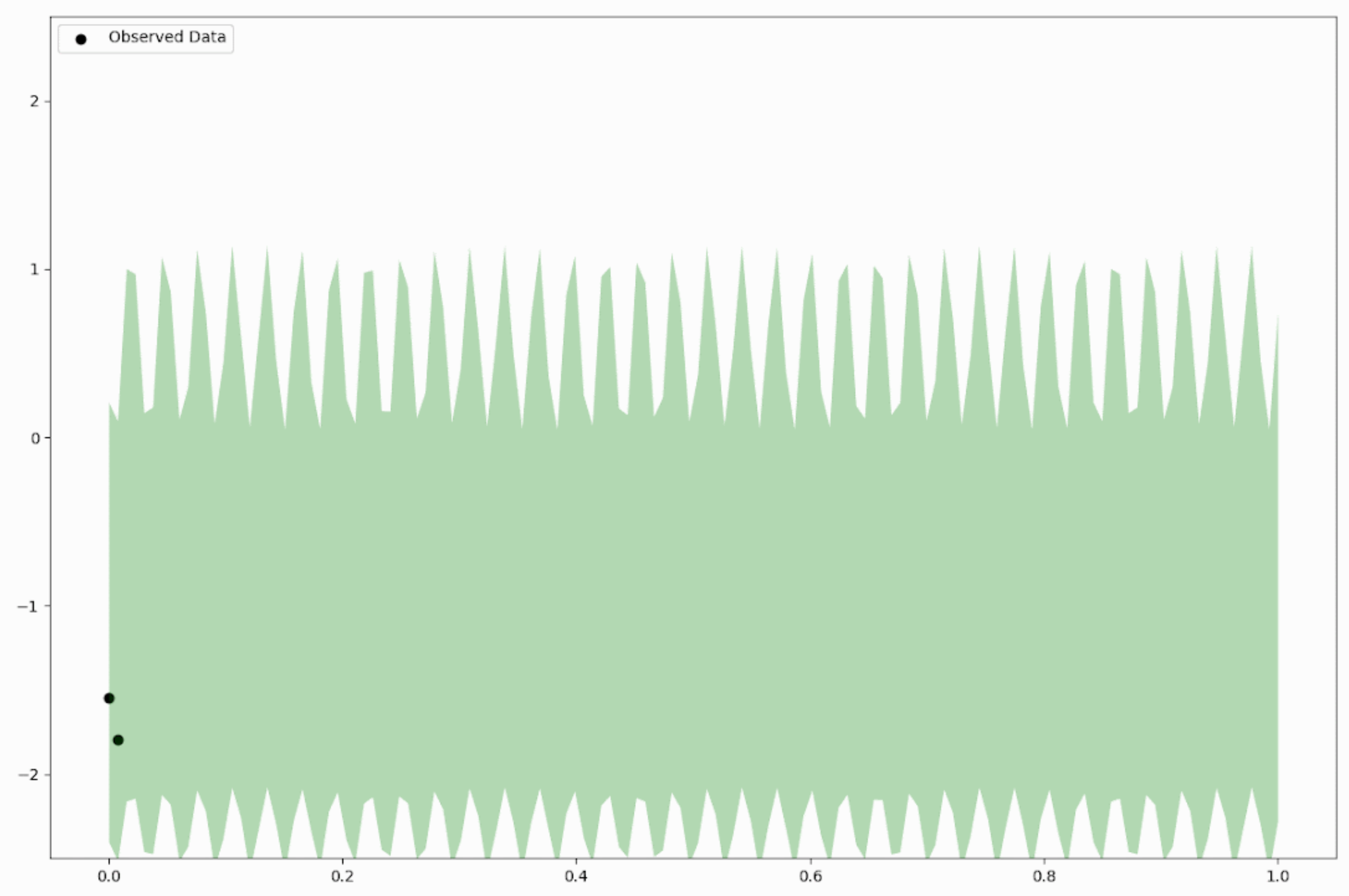
Morphology-Phonology Interactions

- Picked this domain to study how effective, productive generalization can happen from tiny amour
- Our task: solve phonolo

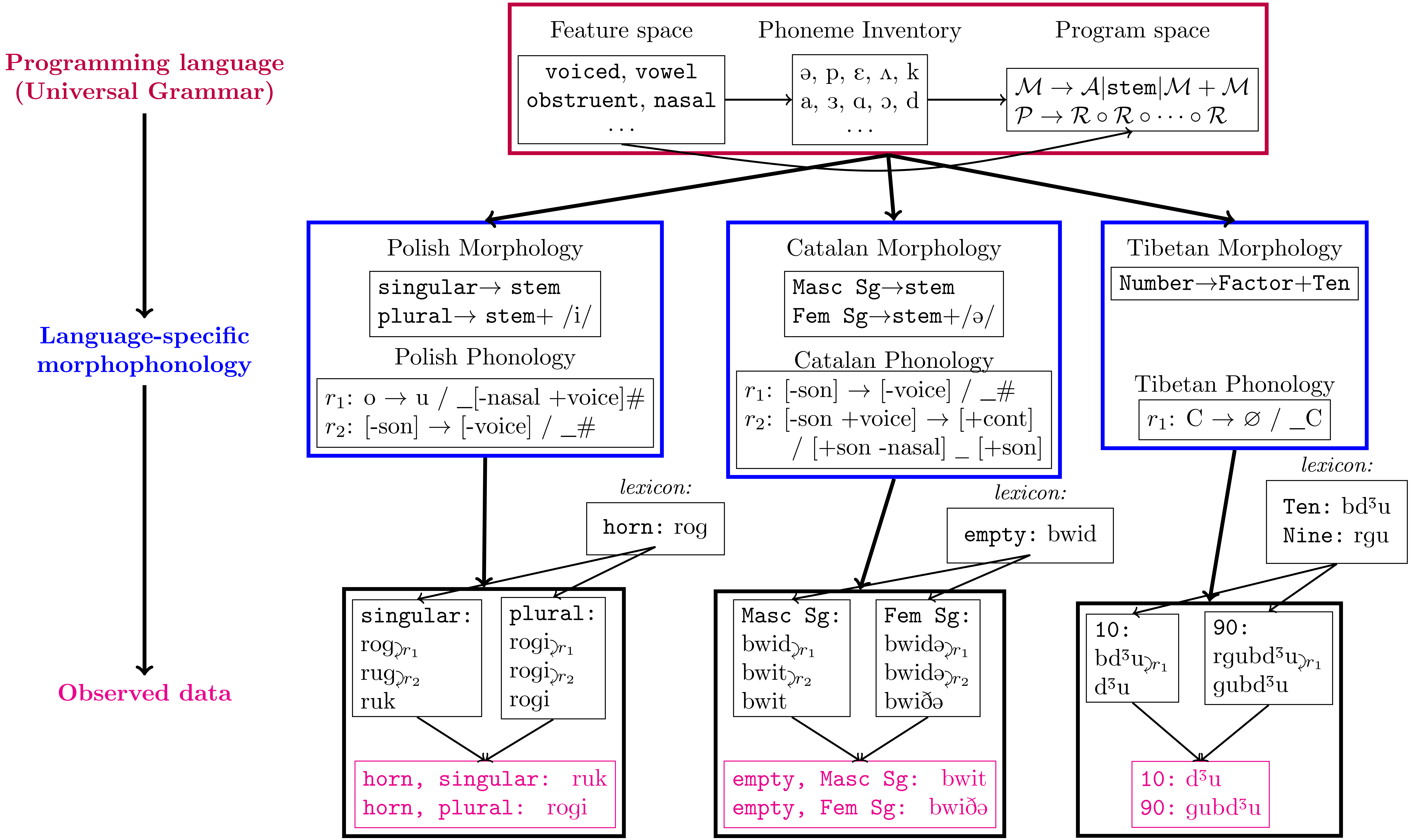
- ~70 problems from 58 languages
- Datasets very small
- Largest phonological dataset of its kind



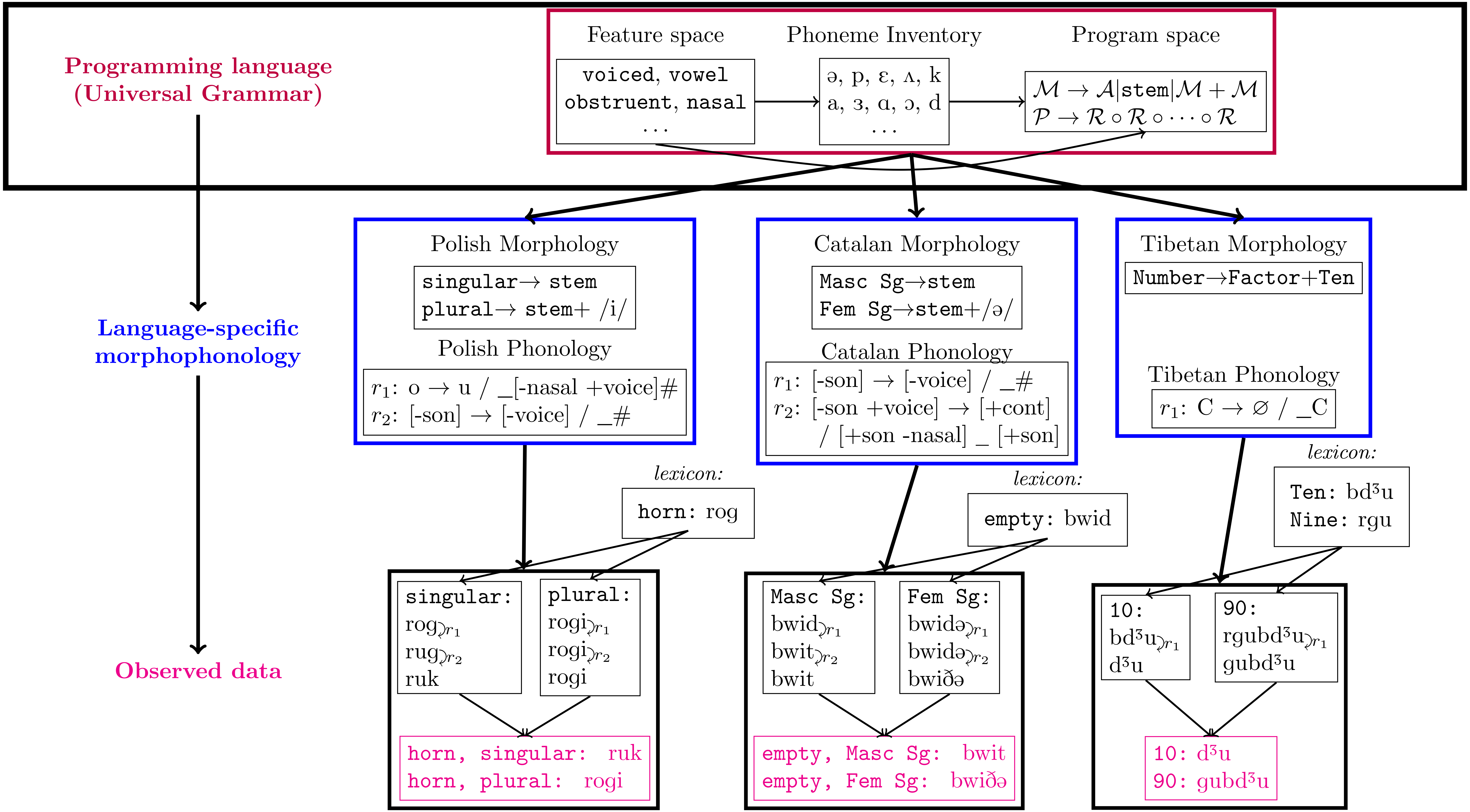
Learning as Bayesian Program Induction




```
@gen function gaussian_process_DSL_model(t::Vector{Float64})

  DSL_program = Periodic(0.6945, 0.0203)

  Σ = [
    [DSL_program(t[i], t[j]) for j=1:length(t)]
    for i=1:length(t)
  ]

  ϵ = 0.6724

  x = Vector{Float64}(undef, length(t))
  x[1] ~ normal(0, Σ[1,1])
  for i=2:length(t)
    μ = Σ[i,1:i-1]' * Σ[1:i-1, 1:i-1]⁻¹ * x[1:i-1]
    σ² = Σ[i,i] .- Σ[i, 1:i-1]' * Σ[1:i-1, 1:i-1]⁻¹ * Σ[i, 1:i-1]
    x[i] ~ normal(μ, σ² + ϵ)
  end

return x
end
```



Programming language
(Universal Grammar)
Language-specific
morphophonology
Observed data
Feature space
voiced, vowel
obstruent, nasal
...
Phoneme Inventory
ə, p, ɛ, ʌ, k
a, ɜ, ɑ, ɔ, d
...
Program space
$\mathcal{M} \to \mathcal{A}|\texttt{stem}|\mathcal{M}+\mathcal{M}$
$\mathcal{P} \to \mathcal{R} \circ \mathcal{R} \circ \cdots \circ \mathcal{R}$
Polish Morphology
singular→ stem
plural→ stem+ /i/
Polish Phonology
r_1: o → u / _[-nasal +voice]#
r_2: [-son] → [-voice] / _#
Catalan Morphology
Masc Sg→stem
Fem Sg→stem+/ə/
Catalan Phonology
r_1: [-son] → [-voice] / _#
r_2: [-son +voice] → [+cont] / [+son -nasal] _ [+son]
Tibetan Morphology
Number→Factor+Ten
Tibetan Phonology
r_1: C → ∅ / _C
lexicon:
horn: rog
lexicon:
empty: bwid
lexicon:
Ten: bd³u
Nine: rgu
singular:
rog
rug
ruk
plural:
rogi
rogi
rogi
Masc Sg:
bwid
bwit
bwit
Fem Sg:
bwidə
bwidə
bwiðə
10:
bd³u
d³u
90:
rgubd³u
gubd³u
horn, singular: ruk
horn, plural: rogi
empty, Masc Sg: bwit
empty, Fem Sg: bwiðə
10: d³u
90: gubd³u


Programming language
(Universal Grammar)
Feature space
voiced, vowel
obstruent, nasal
...
Phoneme Inventory
ə, p, ɛ, ʌ, k
a, ɜ, ɑ, ɔ, d
...
Program space
$\mathcal{M} \rightarrow \mathcal{A} | \texttt{stem} | \mathcal{M} + \mathcal{M}$
$\mathcal{P} \rightarrow \mathcal{R} \circ \mathcal{R} \circ \cdots \circ \mathcal{R}$
Language-specific
morphophonology
Polish Morphology
singular→ stem
plural→ stem+ /i/
Polish Phonology
r_1: o → u / _[-nasal +voice]#
r_2: [-son] → [-voice] / _#
Catalan Morphology
Masc Sg→stem
Fem Sg→stem+/ə/
Catalan Phonology
r_1: [-son] → [-voice] / _#
r_2: [-son +voice] → [+cont]
/ [+son -nasal] _ [+son]
Tibetan Morphology
Number→Factor+Ten
Tibetan Phonology
r_1: C → ∅ / _C
lexicon:
horn: rog
lexicon:
empty: bwid
lexicon:
Ten: bd³u
Nine: rgu
singular:
rog r_1
rug r_2
ruk
plural:
rogi r_1
rogi r_2
rogi
Masc Sg:
bwid r_1
bwit r_2
bwit
Fem Sg:
bwidə r_1
bwidə r_2
bwiðə
10:
bd³u r_1
d³u
90:
rgubd³u r_1
gubd³u
Observed data
horn, singular: ruk
horn, plural: rogi
empty, Masc Sg: bwit
empty, Fem Sg: bwiðə
10: d³u
90: gubd³u

Programming Language

- Morphology: Simple concatenative rules combining underlying forms of morphemes based on morphological function.

  FUNCTION 1: # *prefix* + stem + #

  FUNCTION 2: # + stem + #

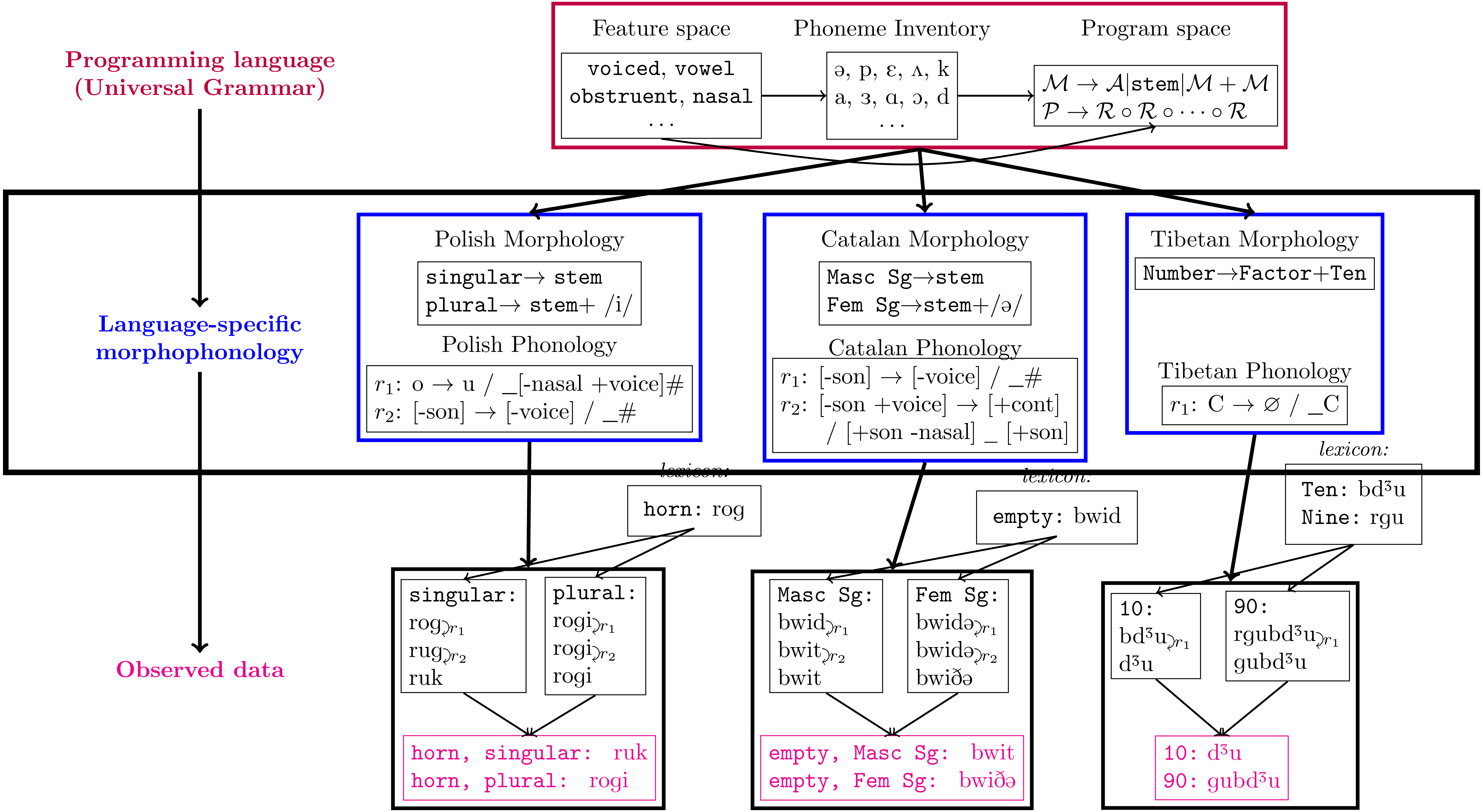
- Phonology: Ordered rules that transform resulting phone sequences.

  input → output / context _ context

Programming Language

Grammar rule	English description
$\mathcal{M} \to \mathcal{M} + \mathcal{M}$	Morphologies $\mathcal{M}$ are concatenations of more basic components
$\mathcal{M} \to \texttt{stem}$	Morphologies can referred to the stem of a lexeme
$\mathcal{M} \to \mathcal{A}$	Morphologies can include constant affixes
$\mathcal{A} \to$ sequence of phonemes	

Grammar rule	English description
$\mathcal{P} \to \mathcal{R} \circ \mathcal{R} \circ \cdots \circ \mathcal{R}$	**P**honology is compositions of **r**ewrites
$\mathcal{R} \to \mathcal{F} \longrightarrow \mathcal{C}/\mathcal{T}\_\mathcal{T}$	Rewrite **f**ocus to **c**hange between **t**riggers
$\mathcal{T} \to \#\mathcal{T}'\vert\mathcal{T}'$	**T**riggers optionally match end of string, $\#$
$\mathcal{T}' \to \epsilon\vert\mathcal{X}\mathcal{T}'\vert\mathcal{X}^*\mathcal{T}'$	**T**riggers are sequences of matrices $\mathcal{X}$
$\mathcal{X} \to a\vert t\vert s\vert \cdots$	Matrices can be constant phonemes
$\mathcal{X} \to [\pm\mathcal{E}\pm\mathcal{E}\cdots\pm\mathcal{E}]$	Matrices check features $\mathcal{E}$
$\mathcal{E} \to \text{voice}\vert\text{nasal}\vert \cdots$	Standard phonological f**e**atures
$\mathcal{F} \to \mathcal{X}$	Focus can be a feature matrix
$\mathcal{F} \to \mathbb{Z}$	Focus can be one of the triggers (copies it)
$\mathcal{F} \to \varnothing$	Insertion rule
$\mathcal{C} \to \mathcal{X}$	Structural change can be a feature matrix
$\mathcal{C} \to \varnothing$	Deletion rule
$\mathcal{C} \to \mathbb{Z}$	Structural change constrained to match a triggering feature matrix


Programming language
(Universal Grammar)
Feature space
voiced, vowel
obstruent, nasal
...
Phoneme Inventory
ə, p, ɛ, ʌ, k
a, ʒ, ɑ, ɔ, d
...
Program space
$\mathcal{M} \rightarrow \mathcal{A}|\texttt{stem}|\mathcal{M}+\mathcal{M}$
$\mathcal{P} \rightarrow \mathcal{R} \circ \mathcal{R} \circ \cdots \circ \mathcal{R}$
Language-specific
morphophonology
Polish Morphology
singular→ stem
plural→ stem+ /i/
Polish Phonology
r_1: o → u / _[-nasal +voice]#
r_2: [-son] → [-voice] / _#
Catalan Morphology
Masc Sg→stem
Fem Sg→stem+/ə/
Catalan Phonology
r_1: [-son] → [-voice] / _#
r_2: [-son +voice] → [+cont] / [+son -nasal] _ [+son]
Tibetan Morphology
Number→Factor+Ten
Tibetan Phonology
r_1: C → ∅ / _C
lexicon:
horn: rog
lexicon:
empty: bwid
lexicon:
Ten: bd^{3}u
Nine: rgu
Observed data
singular:
rog$_{r_1}$
rug$_{r_2}$
ruk
plural:
rogi$_{r_1}$
rogi$_{r_2}$
rogi
horn, singular: ruk
horn, plural: rogi
Masc Sg:
bwid$_{r_1}$
bwit$_{r_2}$
bwit
Fem Sg:
bwidə$_{r_1}$
bwidə$_{r_2}$
bwiðə
empty, Masc Sg: bwit
empty, Fem Sg: bwiðə
10:
bd^{3}u$_{r_1}$
d^{3}u
90:
rgubd^{3}u$_{r_1}$
gubd^{3}u
10: d^{3}u
90: gubd^{3}u

Use program synthesis techniques

SAT/SMT solvi

✓ Guarantee: Exact optimization

✗ No guarantee: runtime

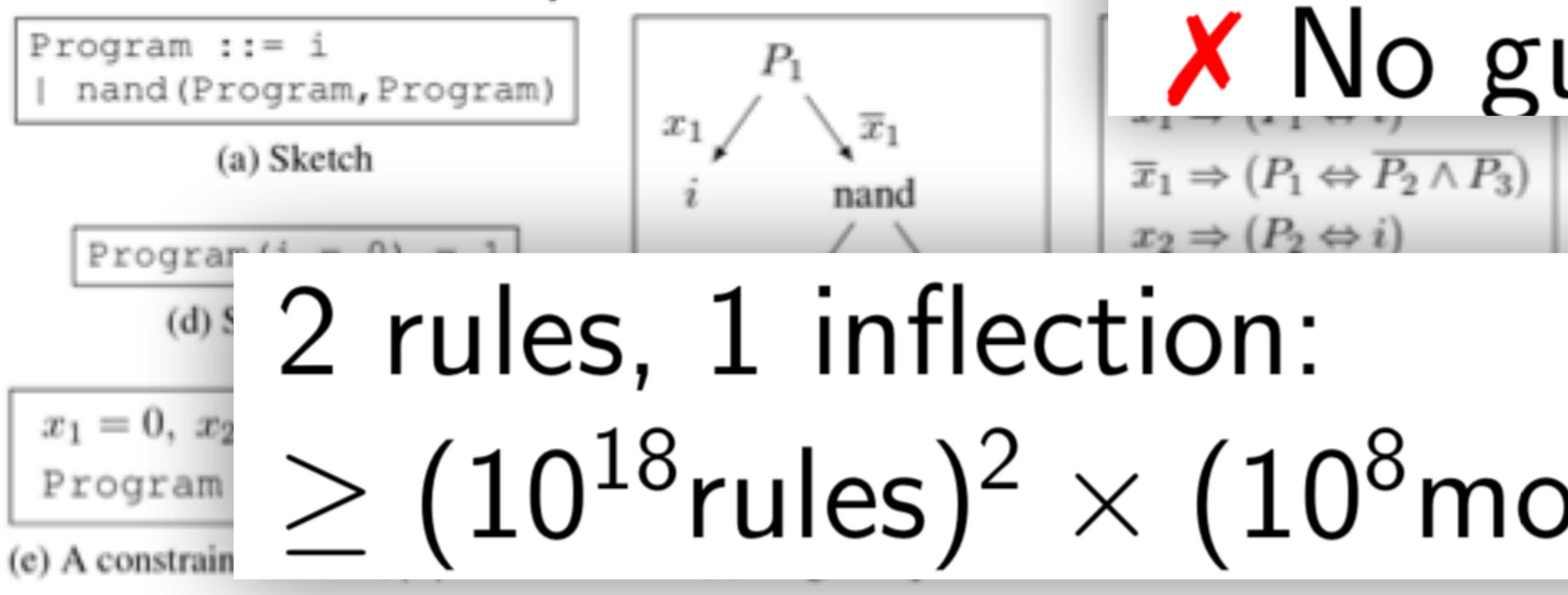
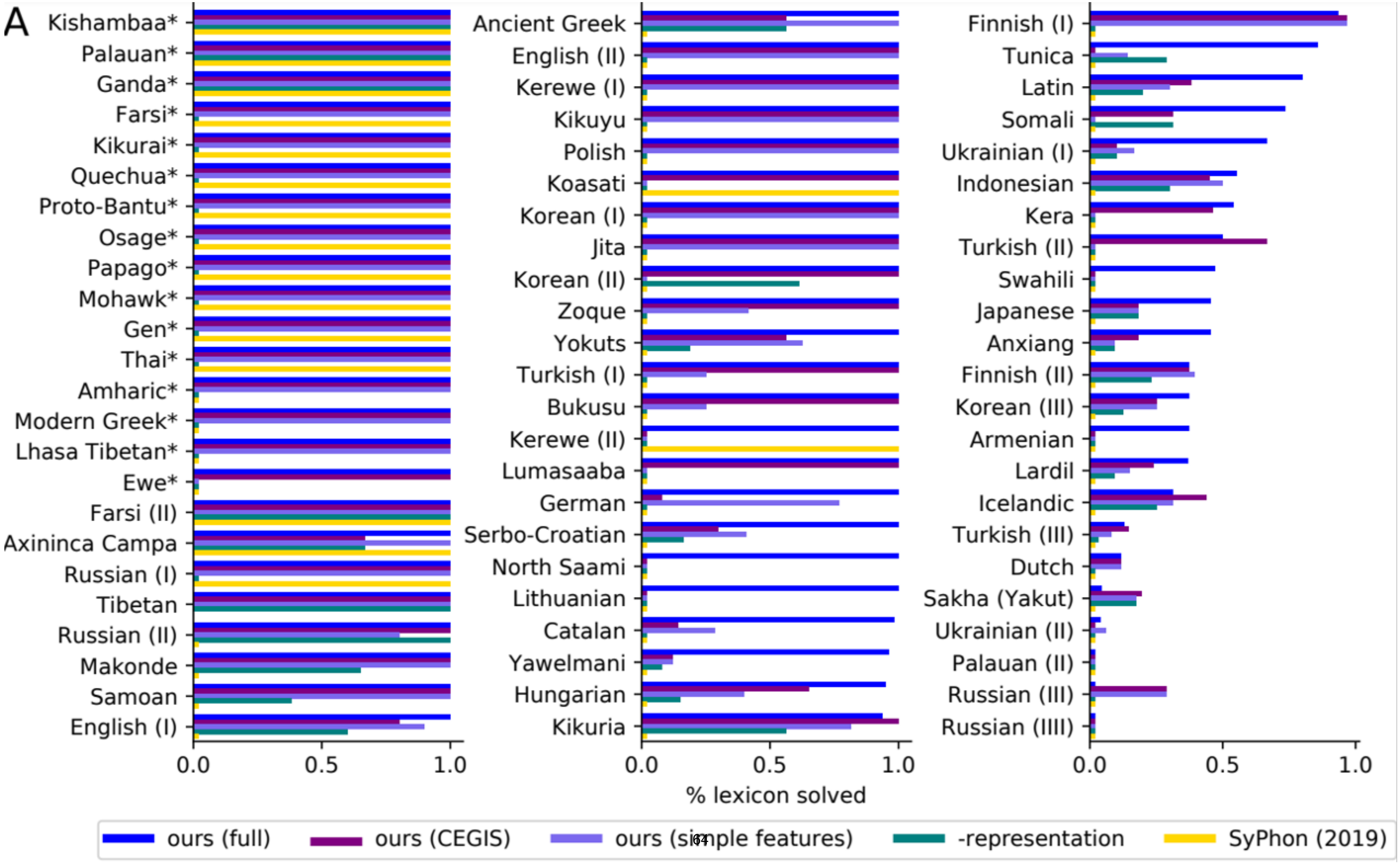


Figure 2: Synthesizing a program via sketching and constraint solving. `Typewriter font` refers to pieces of programs or sketches, while math font refers to pieces of a constraint satisfaction problem. The variable `i` is the program input.

2 rules, 1 inflection:

$\geq (10^{18}\text{rules})^2 \times (10^8\text{morphologies}) = 10^{42}\text{models}$

```
    if(t == 0){return x;}

    if(t == 3){return a * b;}
    if(t == 4){return a + b;}
    if(t == 5){return a - b;}
}
harness void sketch( int x, int y, int z ){
    assert rec(x,y, z) == (x + x) * (y - z);
}
```


A
Kishambaa*
Palauan*
Ganda*
Farsi*
Kikurai*
Quechua*
Proto-Bantu*
Osage*
Papago*
Mohawk*
Gen*
Thai*
Amharic*
Modern Greek*
Lhasa Tibetan*
Ewe*
Farsi (II)
Axininca Campa
Russian (I)
Tibetan
Russian (II)
Makonde
Samoan
English (I)
Ancient Greek
English (II)
Kerewe (I)
Kikuyu
Polish
Koasati
Korean (I)
Jita
Korean (II)
Zoque
Yokuts
Turkish (I)
Bukusu
Kerewe (II)
Lumasaaba
German
Serbo-Croatian
North Saami
Lithuanian
Catalan
Yawelmani
Hungarian
Kikuria
Finnish (I)
Tunica
Latin
Somali
Ukrainian (I)
Indonesian
Kera
Turkish (II)
Swahili
Japanese
Anxiang
Finnish (II)
Korean (III)
Armenian
Lardil
Icelandic
Turkish (III)
Dutch
Sakha (Yakut)
Ukrainian (II)
Palauan (II)
Russian (III)
Russian (IIII)
0.0
0.5
1.0
% lexicon solved
ours (full)
ours (CEGIS)
ours (simple features)
-representation
SyPhon (2019)

Implications

- Most successful phonological rule learner published to date.
- In most cases, the model finds a correct analysis (i.e., consistent with linguistic analyses).
- Does so from “small data.”
- Many cases it gets partial solutions (not unlike students doing these problems sets).

Outline

- **Productivity**

  Synthesizing Theories of Human Language with Bayesian Program Induction

  Evaluating Distributional Distortion in Neural Language Modeling

- **Compositionality and Incremental Processing**

  Particle Filtering as a Model of Incremental Grounded Sentence Understanding

  The Plausibility of Sampling as an Algorithmic Theory of Sentence Processing

Outline

- **Productivity**

  Synthesizing Theories of Human Language with Bayesian Program Induction

  Evaluating Distributional Distortion in Neural Language Modeling

- **Compositionality and Incremental Processing**

  Particle Filtering as a Model of Incremental Grounded Sentence Understanding

  The Plausibility of Sampling as an Algorithmic Theory of Sentence Processing

Outline

- **<u>Compositionality and Incremental Sentence Processing</u>**

  Particle Filtering as a Model of Incremental Grounded Sentence Understanding

  - Ben Lebrun, Amanda Doucette, Vikash Mansinghka, and Josh Tenenbaum

  The Plausibility of Sampling as an Algorithmic Theory of Sentence Processing

  - Jacob Hoover, Morgan Sonderegger, and Steve Piantadosi

Outline

- **<u>Compositionality and Incremental Sentence Processing</u>**

  Particle Filtering as a Model of Incremental Grounded Sentence Understanding

  - Ben Lebrun, Amanda Doucette, Vikash Mansinghka, and Jo

  The Plausibility of Sampling as an Algorithmic Theory of Sent

  - Jacob Hoover, Morgan Sonderegger, and Steve Piantadosi

Outline

- **<u>Compositionality and Incremental Sentence Processing</u>**

    Particle Filtering as a Model of Incremental Grounded Sentence Understanding

    - Ben Lebrun, Amanda Doucette, Vikash Mansinghka, and Josh Tenenbaum

    The Plausibility of Sampling as an Algorithmic Theory of Sentence Processing

    - Jacob Hoover, Morgan Sonderegger, and Steve Piantadosi

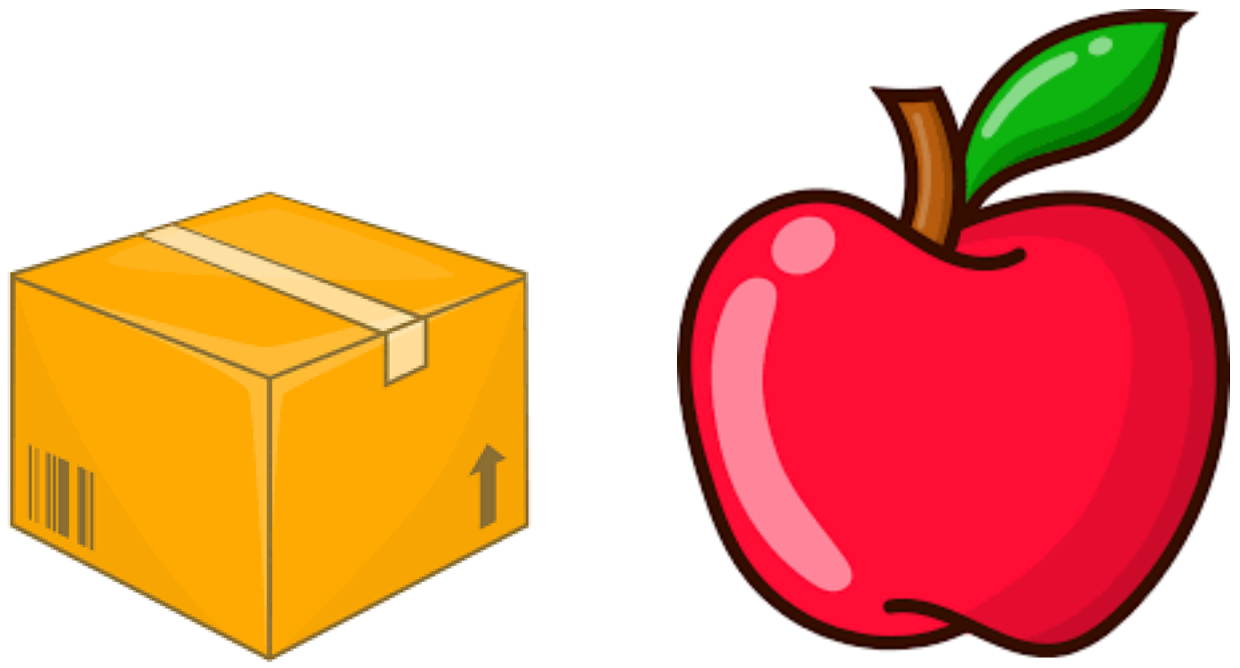
Compositionality and Incremental Processing

- Compositionality: The meaning of sentences is built up from the meaning of words, and the way they are combined.

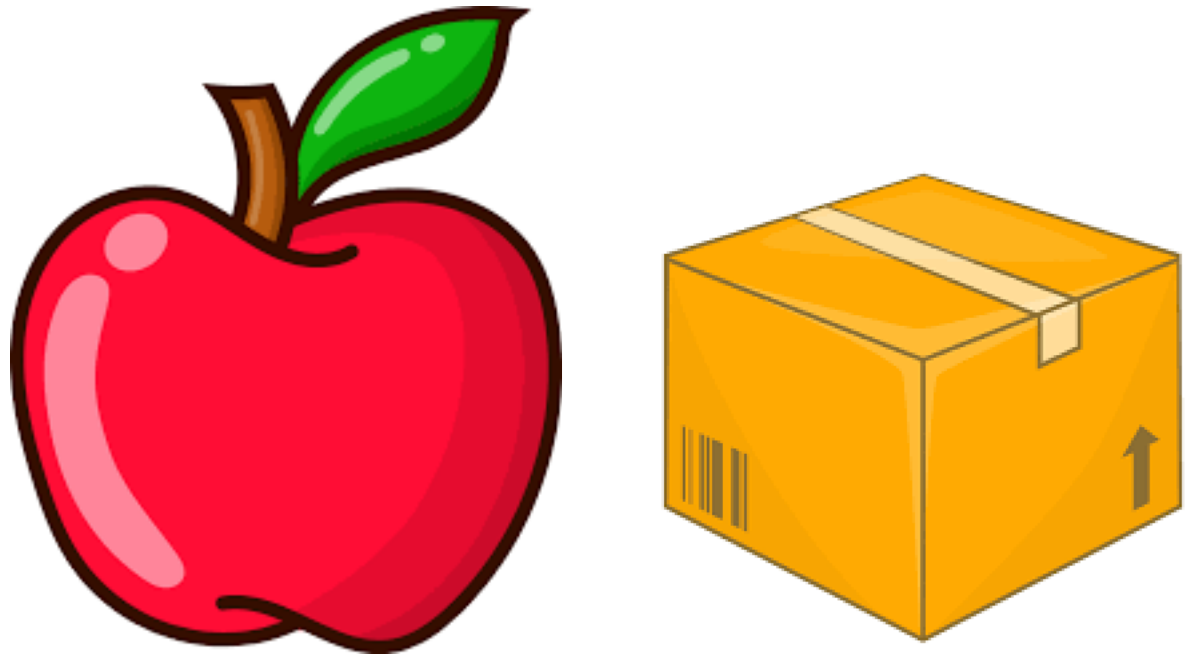
Compositionality and Incremental Processing

- Compositionality: The meaning of sentences is built up from the meaning of words, and the way they are combined.

an apple to the right of a box



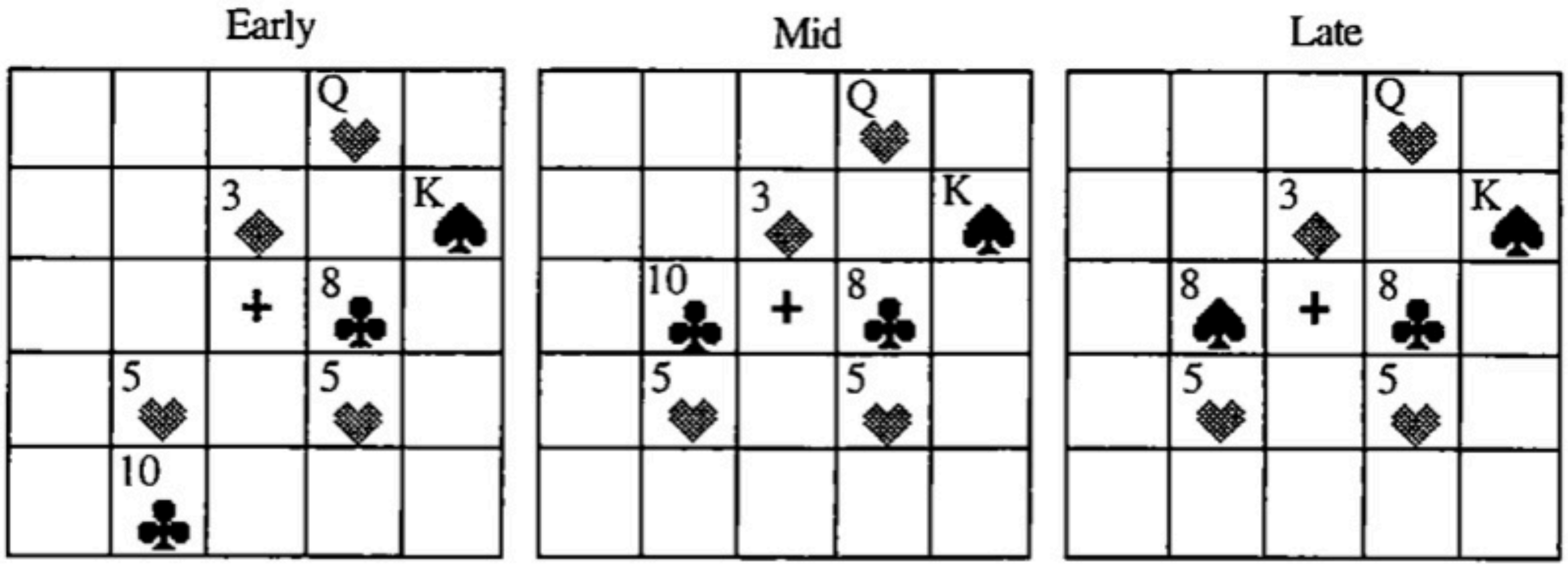
a box to the right of an apple



Compositionality and Incremental Processing

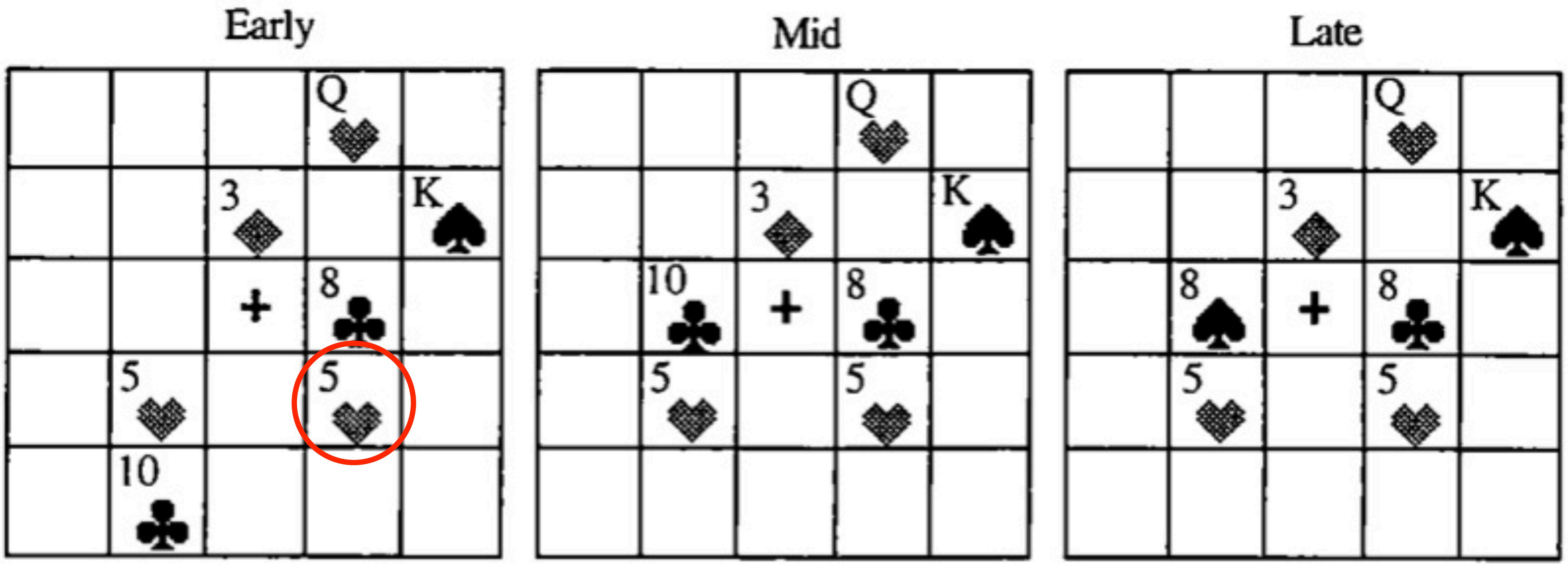
- Compositionality: The meaning of an utterance is a function of the meaning of its parts and the ways they are put together.
- Incrementality: We interpret words as soon as we hear them with as much information as is available at the moment.
    - Human sentence processing is *eager*.
    - Rapidly integrate:
        - Perceptual information.
        - Linguistic knowledge.
        - Prior beliefs.

Eberhard et al. 1995



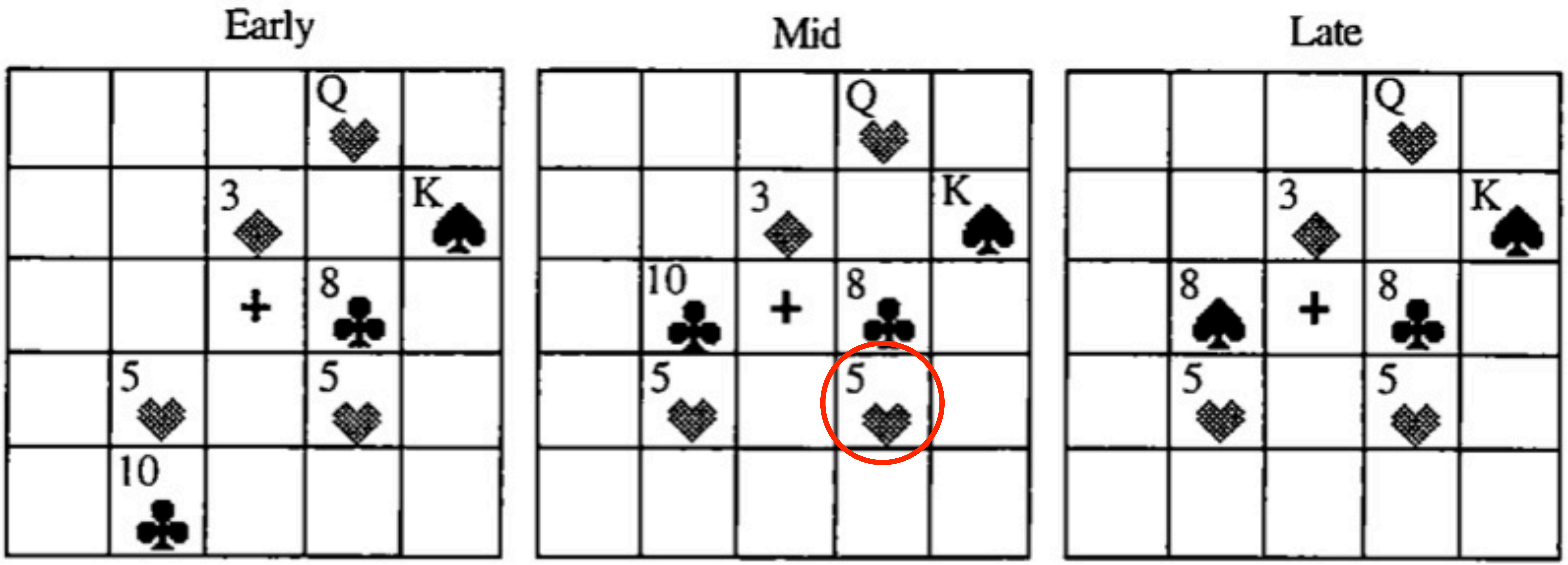
Put the five of hearts that is below the eight of clubs above the three of diamonds

Eberhard et al. 1995



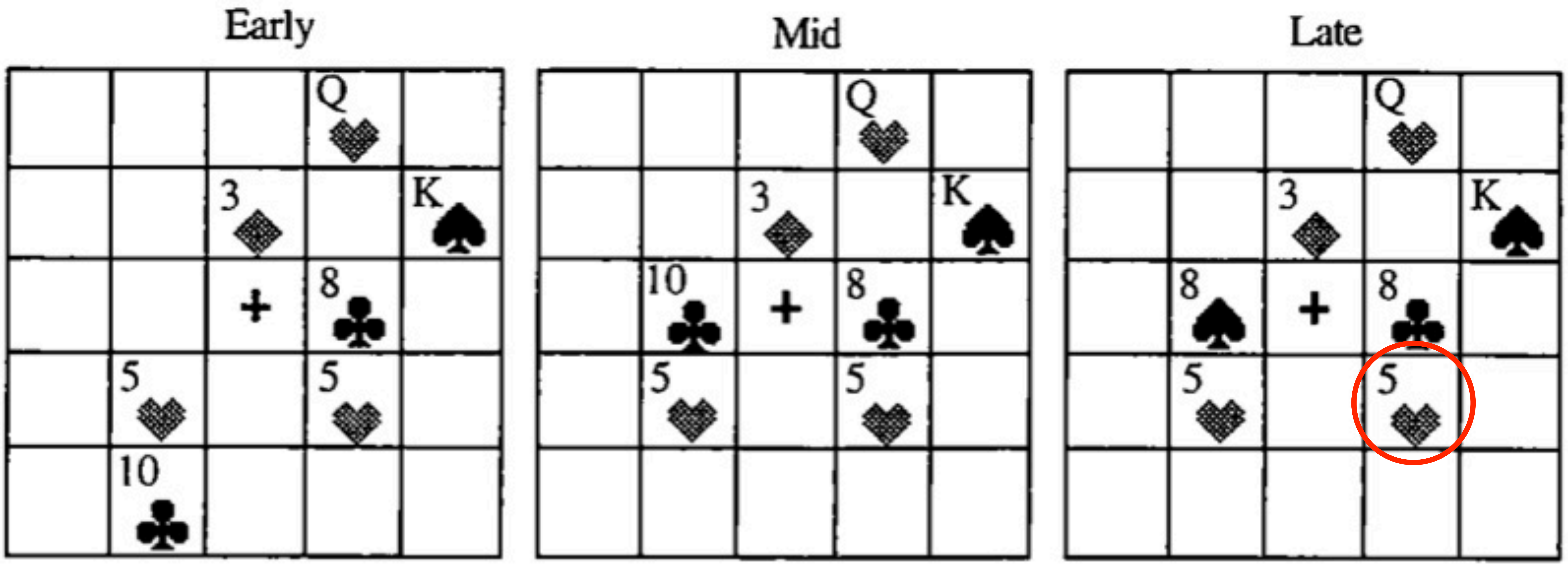
Put the five of hearts that is below the eight of clubs above the three of diamonds

Eberhard et al. 1995



Put the five of hearts that is below the eight of clubs above the three of diamonds

Eberhard et al. 1995



Put the five of hearts that is below the eight of clubs above the three of diamonds

Visually Grounded Language Models

Current models

- Preceding examples involve **visual grounding** of linguistic meaning.
    - Linking of meaning to visual information.
- Much modeling work over the last few years in this domain with many successes.
- Second most important class of AI models (diffusion models).
    - DALL-E
    - StableDiffusion

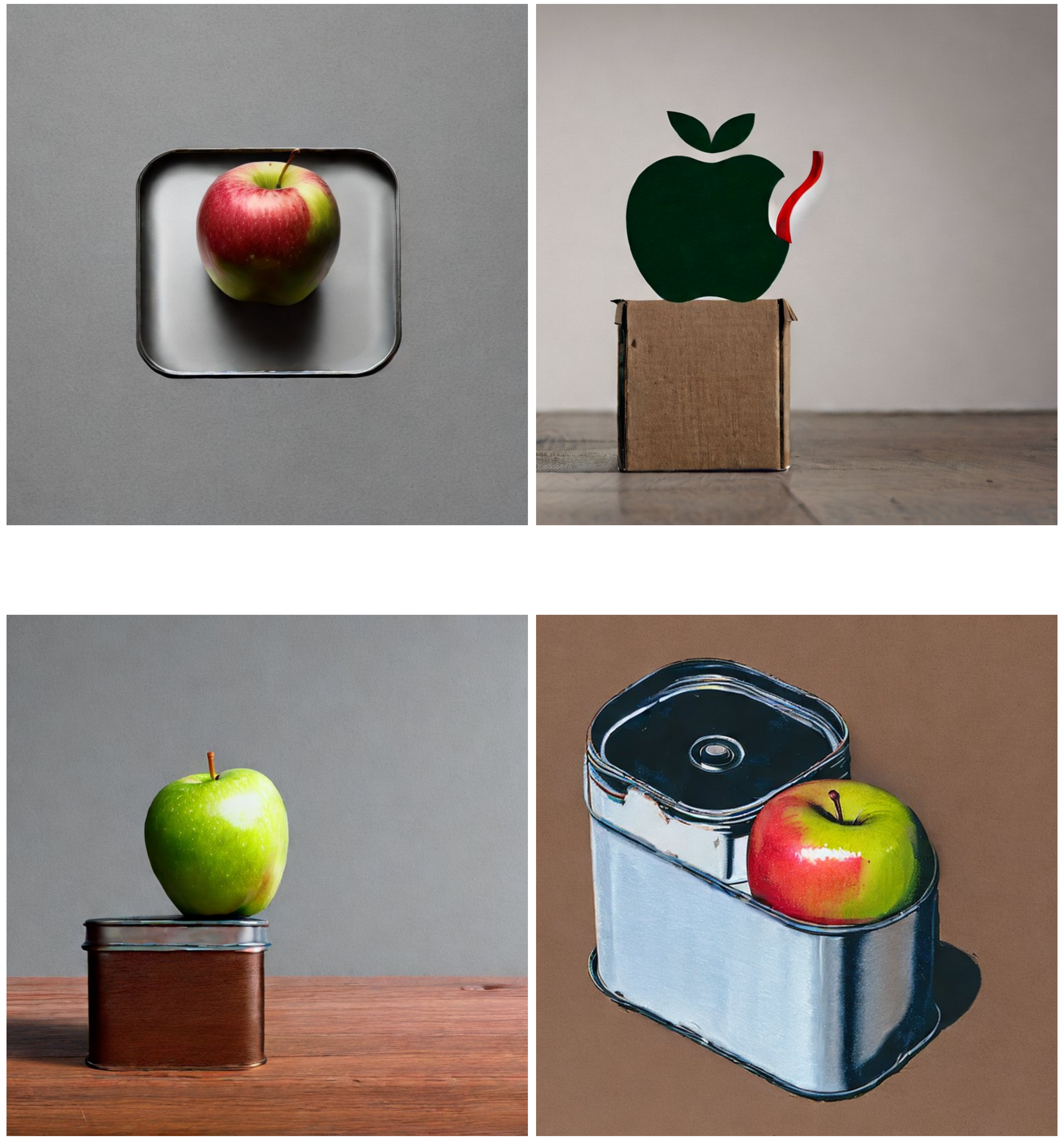
A photo of an astronaut riding a horse.

DALL-E

An apple on top of a box to the left of a can.

DALL-E 2

Stable Diffusion



Visually Grounded Language Models

Current models

- Current models are not (sufficiently) compositional (even very large ones).
- No models of incremental grounding/interpretation.

Probabilistic Neurosymbolic Approach

Overview

- Define a joint probability distribution on utterances and scenes.
- Several intermediate representations.
- Symbolic representations + neural inference components.

Probabilistic Neurosymbolic Approach

Components

Find the can behind…

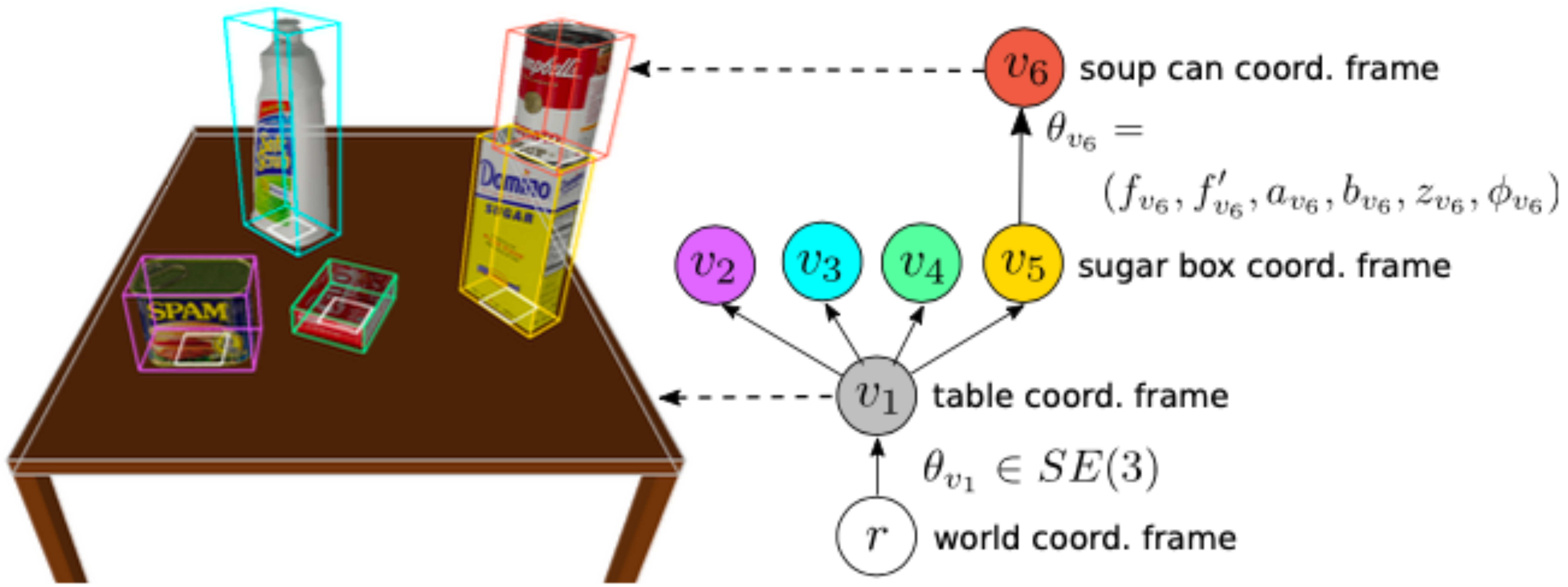
Probabilistic Neurosymbolic Approach

Components

1. Model of visual perception.
    - 3DP3: Parse visual scenes into **scene graphs**.

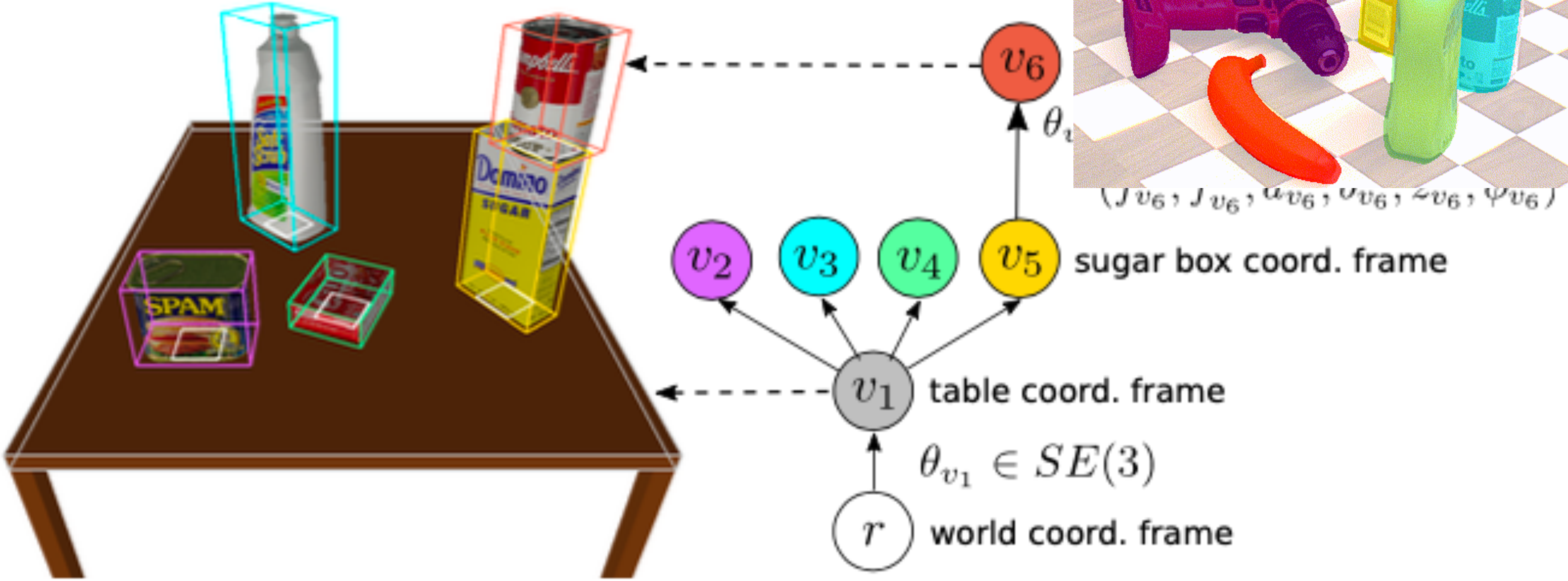
3DP3: 3D Scene Perception via Probabilistic Programming

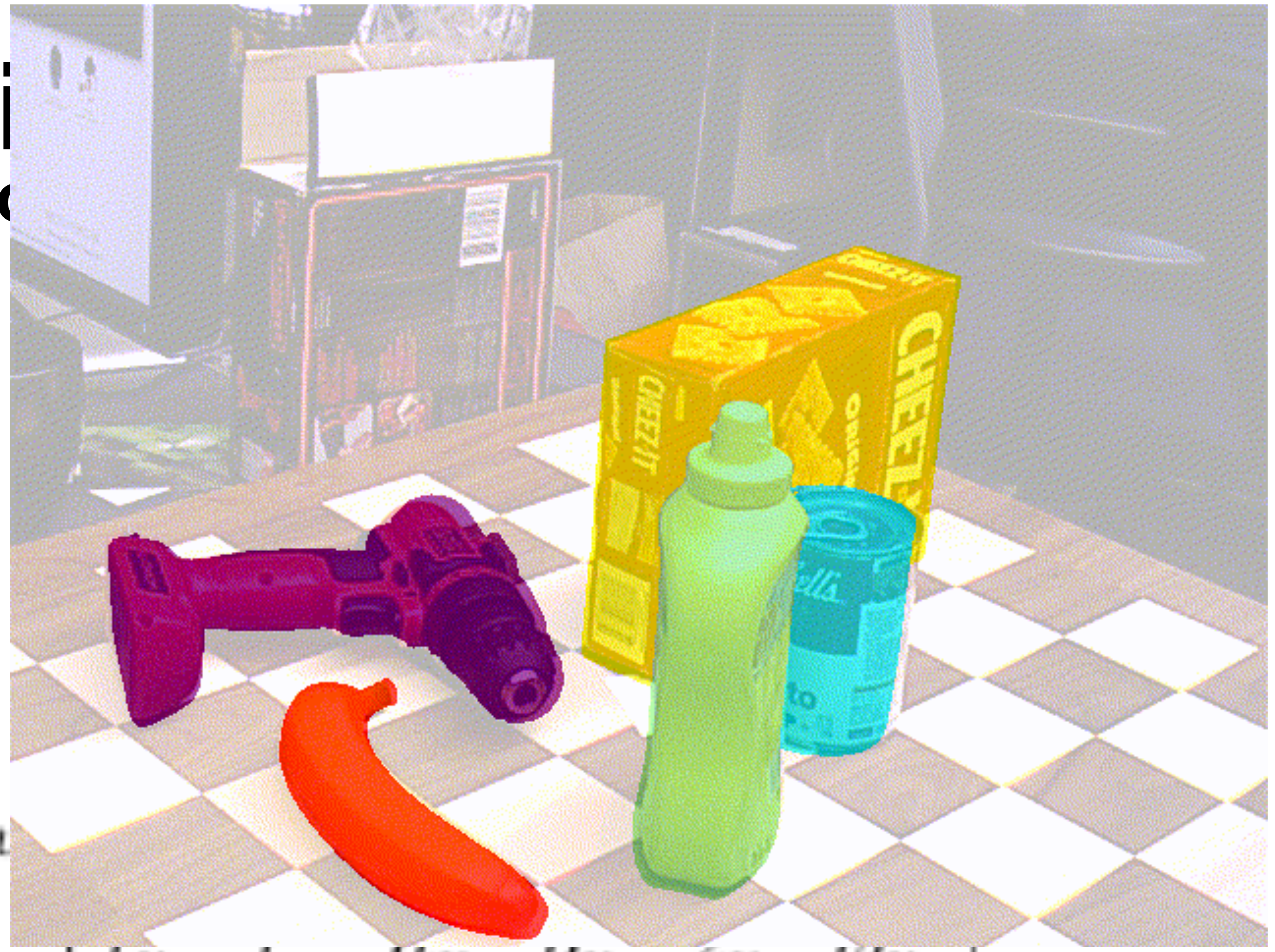
MIT Probabilistic Computing Project (Gothoskar et al 2021)



3DP3: 3D Scene Perception via Probabi

MIT Probabilistic Computing Project (G

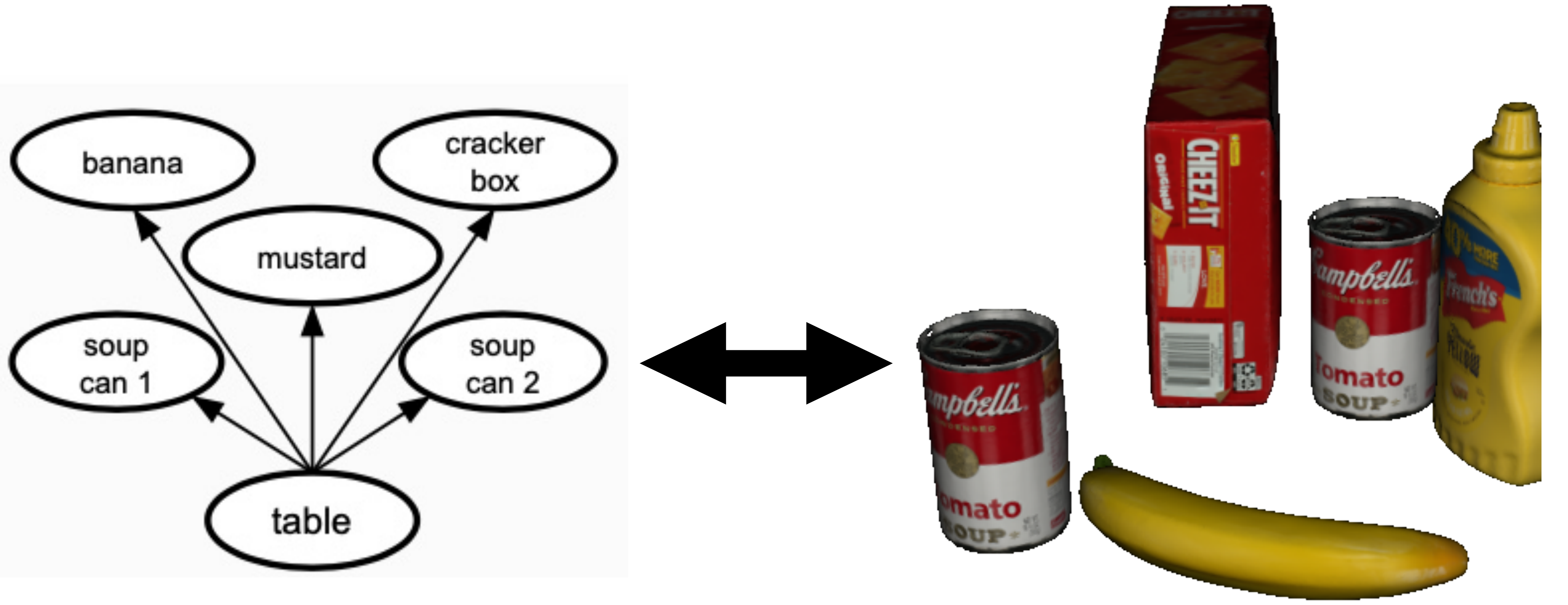




Probabilistic Neurosymbolic Approach

Components

Find the can behind…



Probabilistic Neurosymbolic Approach

Components

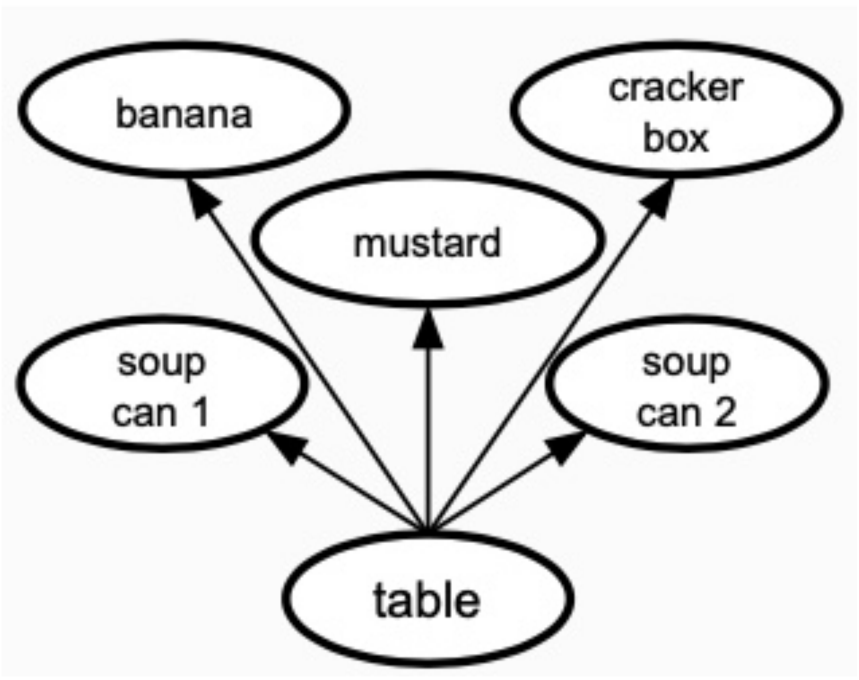
1. Model of visual perception.
   - 3DP3: Parse visual scenes into **scene graphs**.
2. Probabilistic logic expressing constraints on scene graphs.
   - Probabilistic denotational semantics.
   - DRS interpreted as an undirected graphical model.

Probabilistic Neurosymbolic Approach

Components

Find the can behind…

```
λx.can(x) ∧
   ∃y.behind(y,x)
```



Probabilistic Neurosymbolic Approach

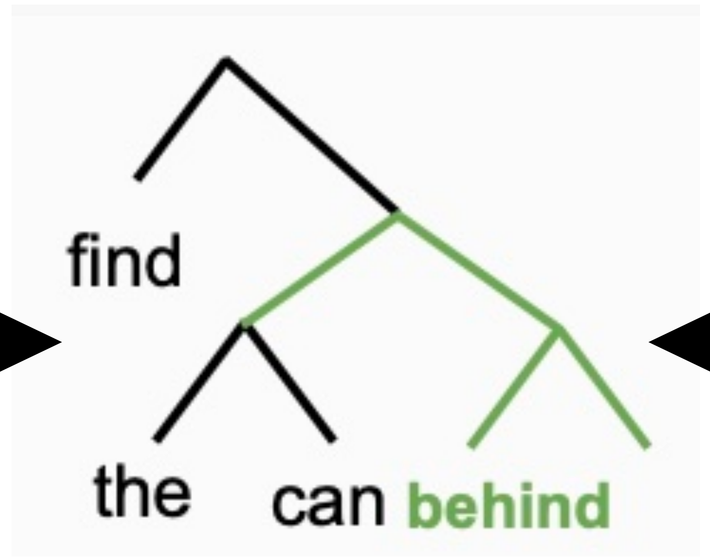
Components

1. Symbolic model of visual perception.
    - 3DP3: Parse visual scenes into **scene graphs**.
2. Probabilistic logic expressing constraints on scene graphs.
    - Probabilistic denotational semantics.
    - DRS interpreted as an undirected graphical model.
3. Parser.
    - Incremental categorial grammar.

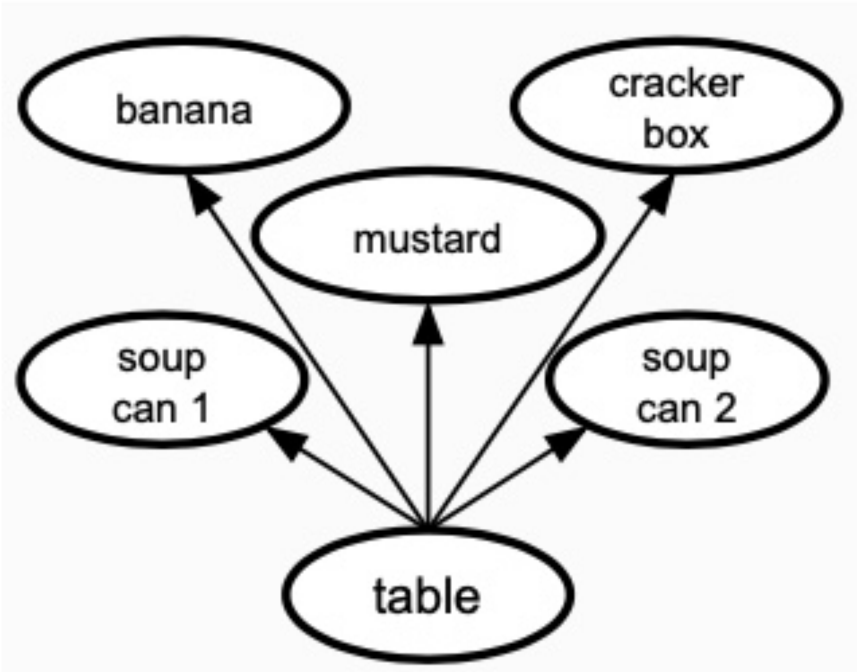
Probabilistic Neurosymbolic Approach

Components

Find the can behind…



$\lambda x.\mathrm{can}(x) \wedge \exists y.\mathrm{behind}(y,x)$



Probabilistic Neurosymbolic Approach

Components

1. Symbolic model of visual perception.
    - 3DP3: Parse visual scenes into **scene graphs**.
2. Probabilistic logic expressing constraints on scene graphs.
    - Probabilistic denotational semantics.
    - DRS interpreted as an undirected graphical model.
3. Parser.
    - Incremental categorial grammar.

Complex joint model

Compositionality

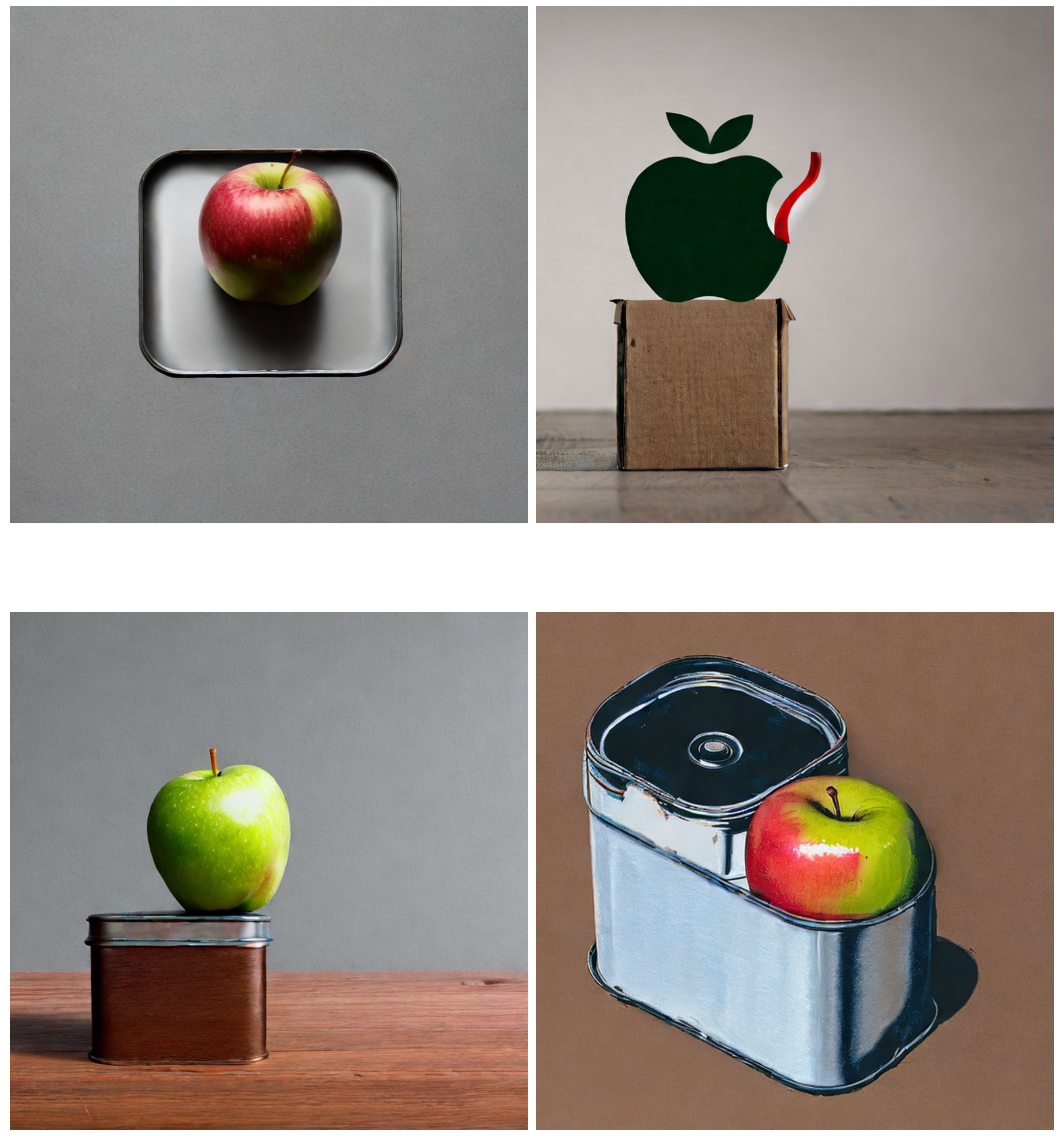
- Condition on a particular utterance and sample scenes.
- Simulates behavior of diffusion models.

Compositionality

An apple on top of a box to the left of a can.

DALL-E

Stable Diffusion



Compositionality

Summary

- Compositional structure is respected categorically (generated images never have incorrect relationship).

- Nevertheless, these models exhibit far less flexibility and coverage than neural models.

Incremental Processing

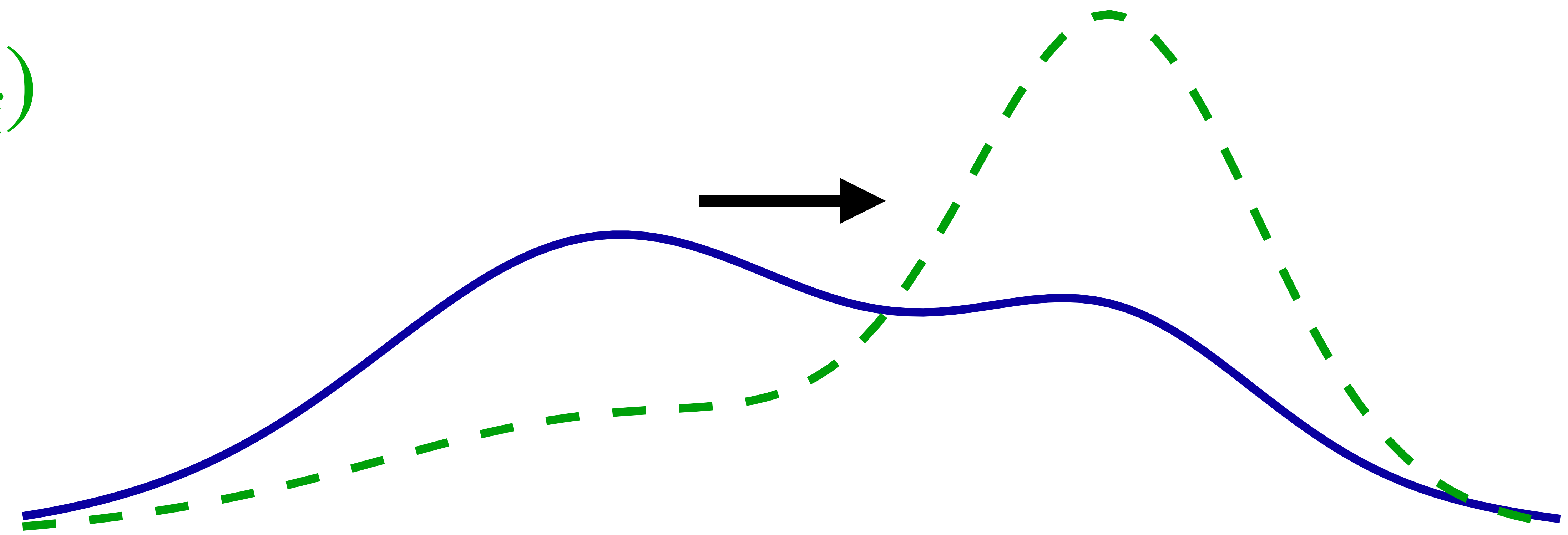
Sequential inference problem

- Sequence of posterior inferences.

$$p(\underbrace{M}_{\text{meaning}} \mid \underbrace{w_1, w_2, \ldots}_{\text{words so far}})$$

- At time step i, observing word w_i leads to some change in belief.

$$p(M \mid w_{<i}) \overset{w_i}{\rightsquigarrow} p(M \mid w_{<i}, w_i)$$



Incremental Inference via Particle Filtering

Sequential Monte Carlo

- This is a complex joint, sequential inference problem.
- How can we do it fast?
- Sample a set of evolving “particles” which contain hypotheses about syntactic structure, meaning, and relationship to a particular scene graph.
- This is a form of sequentialized *Importance Sampling.*

Scene

find

Incremental Processing

Early Disambiguation

Mid Disambiguation

Late Disambiguation

(A) Posterior on referents across three situations with varying points of disambiguation

find the mug on the right of the box

(B) Incrementally observed sentence

Incremental Inference via Particle Filtering

Advantages and problems

- Initial experiments indicate this can work reasonably fast in small cases.
    - Levels of representation mutually constrain one another (avoid massive search).
- <u>Question:</u> Can this serve as a concrete model of human sentence processing?

Outline

- **<u>Incremental Processing</u>**

Particle Filtering as a Model of Incremental Grounded Sentence Processing.

- Ben Lebrun, and Vikash Mansinghka

The Plausibility of Sampling as an Algorithmic Theory of Sentence Processing

- Jacob Hoover, Morgan Sonderegger, and Steve Piantadosi

Outline

- **<u>Incremental Processing</u>**

Particle Filtering as a Model of Incremental Grounded Sentence Processing.

- Ben Lebrun, and Vikash Mansinghka

The Plausibility of Sampling as an Algorithmic Theor

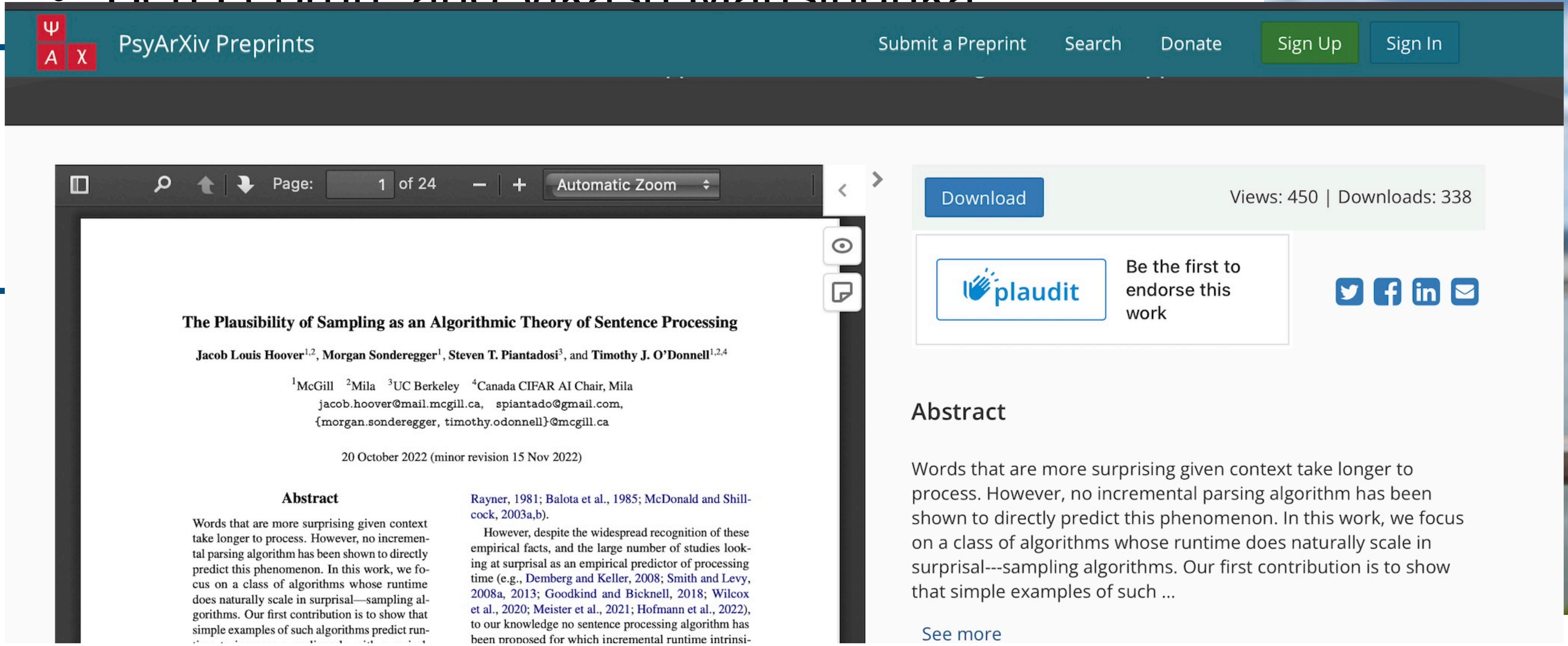
- Jacob Hoover, Morgan Sonderegger, and Steve F

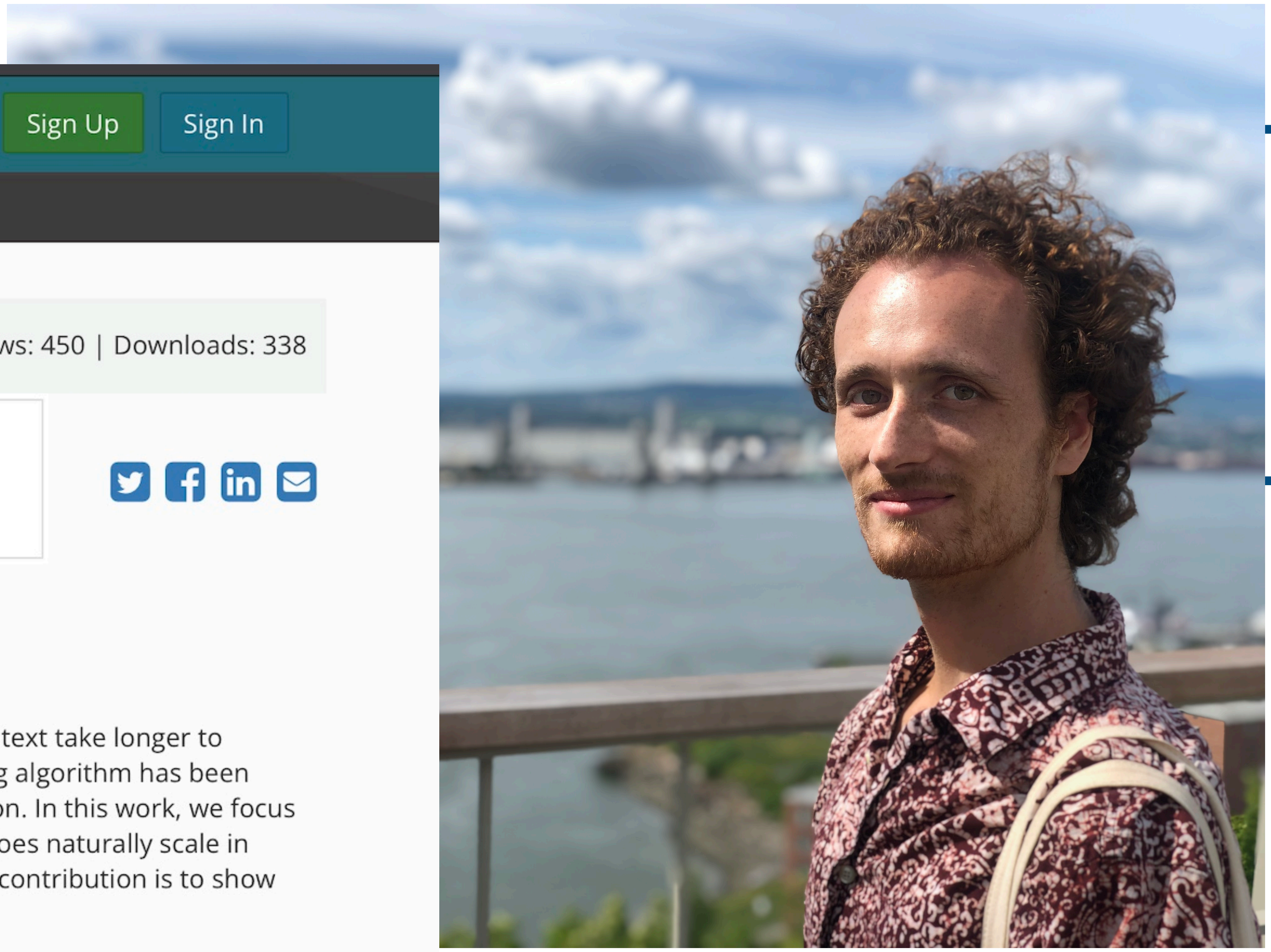
Outline

- **Incremental Processing**

Particle Filtering as a Model of Incremental Grounded Sentence Processing.

- Ben Lebrun, and Vikash Mansinghka





Human Sentence Processing

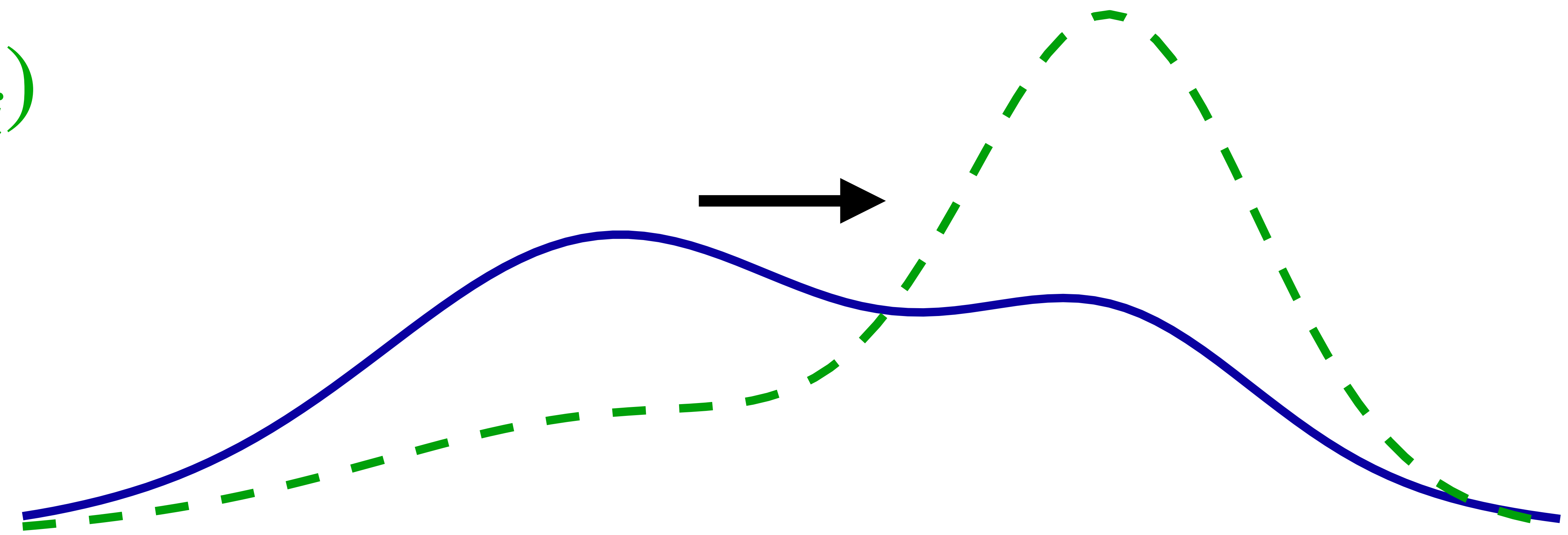
Iterative inference problem

- Sequence of posterior inferences.

$$p(\underbrace{M}_{\text{meaning}} \mid \underbrace{w_1, w_2, \ldots}_{\text{words so far}})$$

- At time step i, observing word w_i leads to some change in belief.

$$p(M \mid w_{<i}) \overset{w_i}{\rightsquigarrow} p(M \mid w_{<i}, w_i)$$

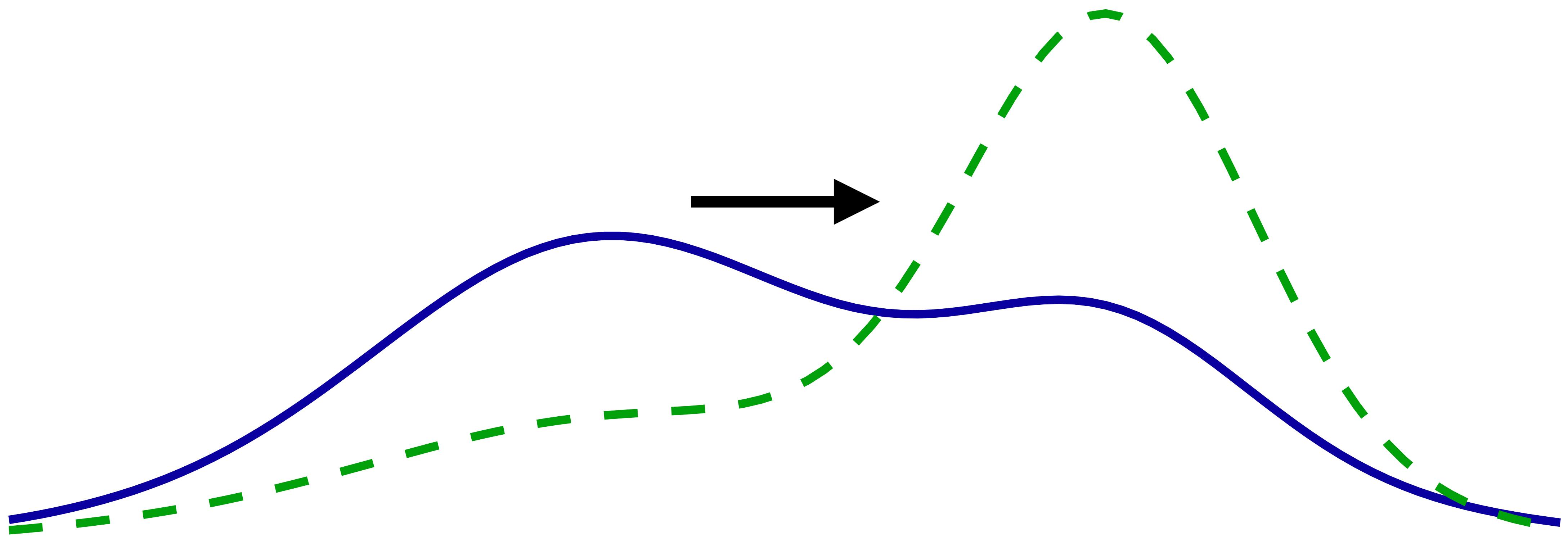


Human Sentence Processing

Effort

- How much work does it take to do this update?

$$p(M \mid w_{<i}) \overset{w_i}{\rightsquigarrow} p(M \mid w_{<i}, w_i)$$



Human Sentence Processing

Surprisal theory

- Empirical fact: Words that are more surprising in context take longer to integrate (i.e., more effort).

- Most common theory risal **Theory** (Hale, 2001).

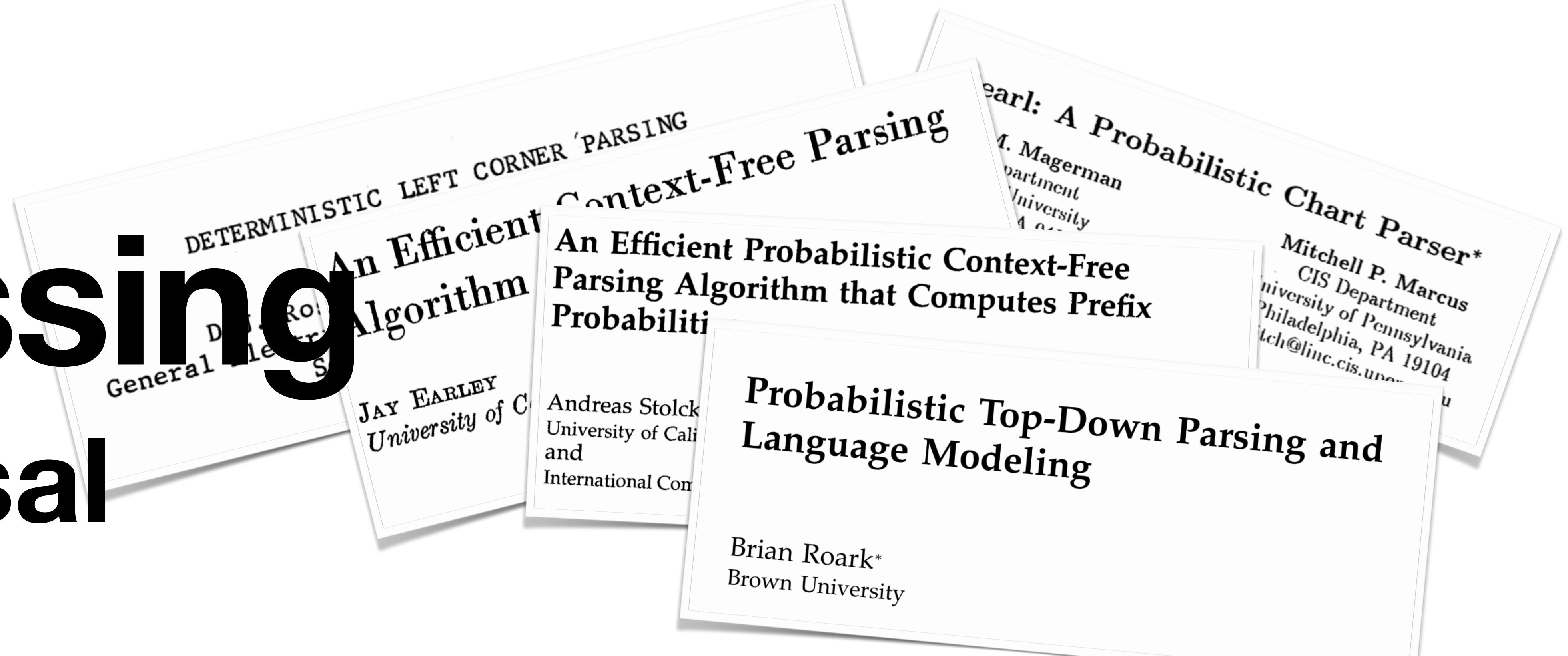
Assumption of linear relationship.

- Effort required to integ tional to its *surprisal*:

$$\text{effort}(w_i) \propto \mathrm{S}(w_i) := -\log p(w_i \mid w_{<i}) = \log \frac{1}{p(w_i \mid w_{<i})}$$

Human Sentence Processing

Algorithms that don't scale in surprisal



- Problem: Most proposed algorithms for incremental processing predict ***no*** relationship with surprisal.
  - Non-probabilistic algorithms (Rosenkrantz and Lewis 1970; Earley 1970)
  - Probabilistic enumerative algorithms (Stolcke 1995; Roark 2001)
  - RNN or Transformer-based models of parsing (Costa 2003; Jin and Schuler 2020; Hu et al. 2021)
  - Causal language models (e.g., LSTM, Transformer-XL, GPT-2/3).

Human Sentence Processing

Algorithms that do scale in surprisal

- Algorithms whose complexity ***does*** scale in surprisal.
  - <u>Importance sampling</u> (Sanz-Alonso, 2016; Chatterjeee & Diaconis, 2017).
    - Special cases include rejection sampling.
    - Assumptions: deterministic likelihood and proposal distribution is the prior (standard assumptions in this literature).
  - <u>Probability-ordered deterministic sequential search</u> (Anderson, 1990; Anderson and Lebiere, 1998)
    - Assumptions: heavy-tailed distributions.

Human Sentence Processing

Algorithms that do scale in surprisal

- Algorithms whose complexity ***does*** scale in surprisal.
  - Importance sampling.
  - Probability-ordered dete[illegible]earch.
- But…!

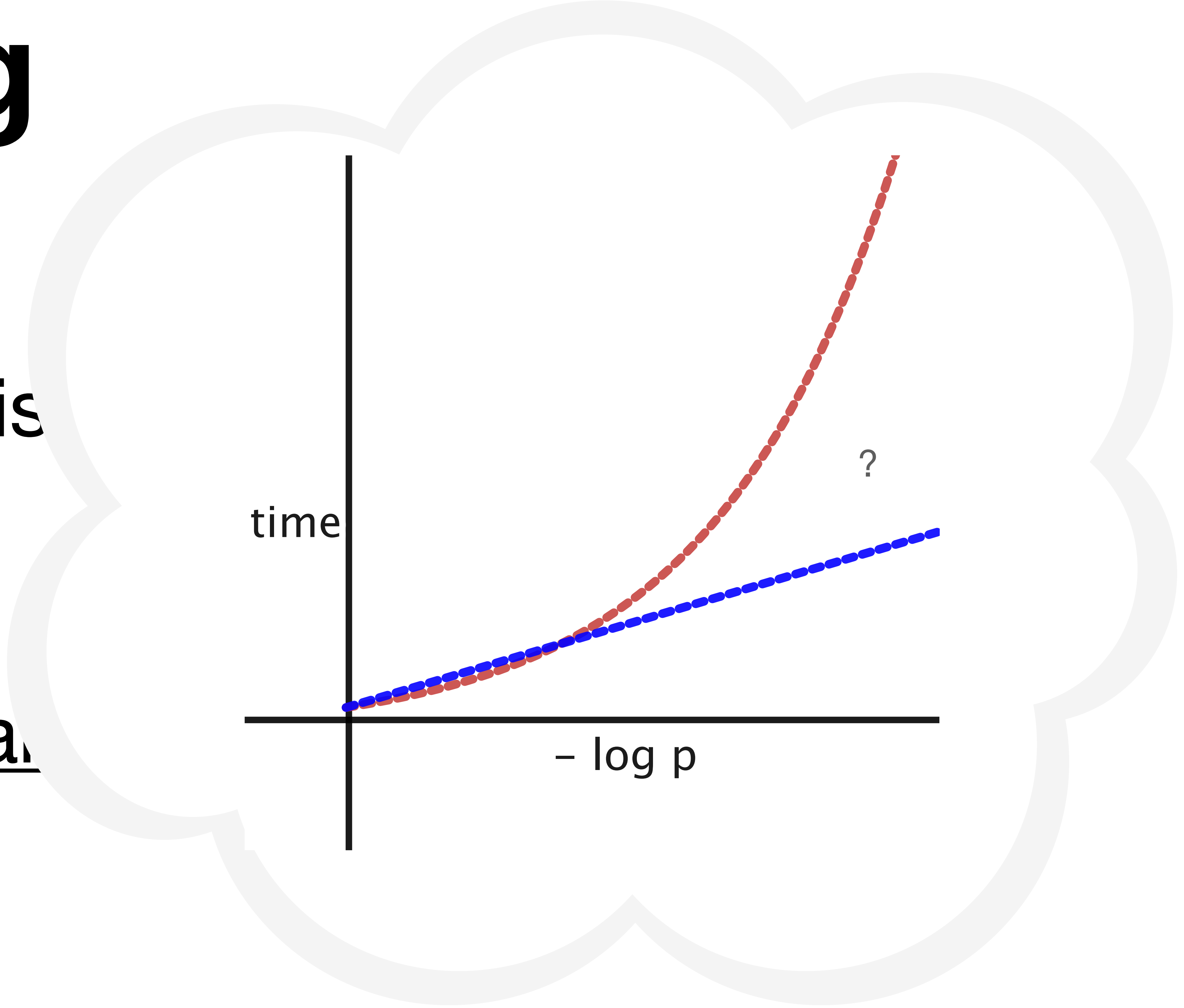
Superlinear relationship!

$$\mathrm{effort}(w_i) \propto e^{\mathrm{S}(w_i)} := e^{-\log p(w_i|w_{<i})} = \frac{1}{p(w_i \mid w_{<i})}$$

Human Sentence Processing

Algorithms that do scale in surprisal

- Algorithms whose complexity ***does*** scale in surpris
    - Importance sampling.
    - Probability-ordered deterministic sequential sea
- But…!



$$\text{effort}(w_i) \propto e^{\mathrm{S}(w_i)} := e^{-\log p(w_i|w_{<i})} = \frac{1}{p(w_i \mid w_{<i})}$$

Human Sent… ce…

Algorithms that … a…

Moreover, sampling theories predict that the variance in effort should also increase as a function of surprisal.

Novel prediction! Differentiates between sampling and sequential search.

- Algorithms whose c… scale in surprisal.
    - Importance sampling.
    - Probability-ordered deterministic sequential search.
- But…!

$$\mathrm{effort}(w_i) \propto e^{\mathrm{S}(w_i)} := e^{-\log p(w_i | w_{<i})} = \frac{1}{p(w_i \mid w_{<i})}$$

Possibilities

1. There is some as yet unknown (at least to me) algorithm that predicts linearity in surprisal.

2. Humans scaling isn't actually linear.

    - E.g. maybe poor surprisal estimates in earlier literature.

3. Surprisal is not the correct quantity to use to predict processing times.

    - Perhaps just correlated with the correct quantity.

Surprisal Theory
Linearity

- Some theoretical arguments in favor of linearity.
    - No process-level proposals.
- Small number of empirical papers argue explicitly for a linear effect of surprisal.

    Smith and Levy, 2008a, 2013; Goodkind and Bicknell, 2018; Wilcox et al., 2020; Hofmann et al., 2022

- Much larger literature simply assumes it.

    Reichle et al., 2003; Dem-berg and Keller, 2008; Boston et al., 2008; Frank, 2009;Roark et al., 2009; Mitchell et al., 2010; Fernandez Mon-salve et al., 2012; Frank et al., 2013; Lowder et al.,2018; Aurnhammer and Frank, 2019; Hao et al., 2020;Merkx and Frank, 2021

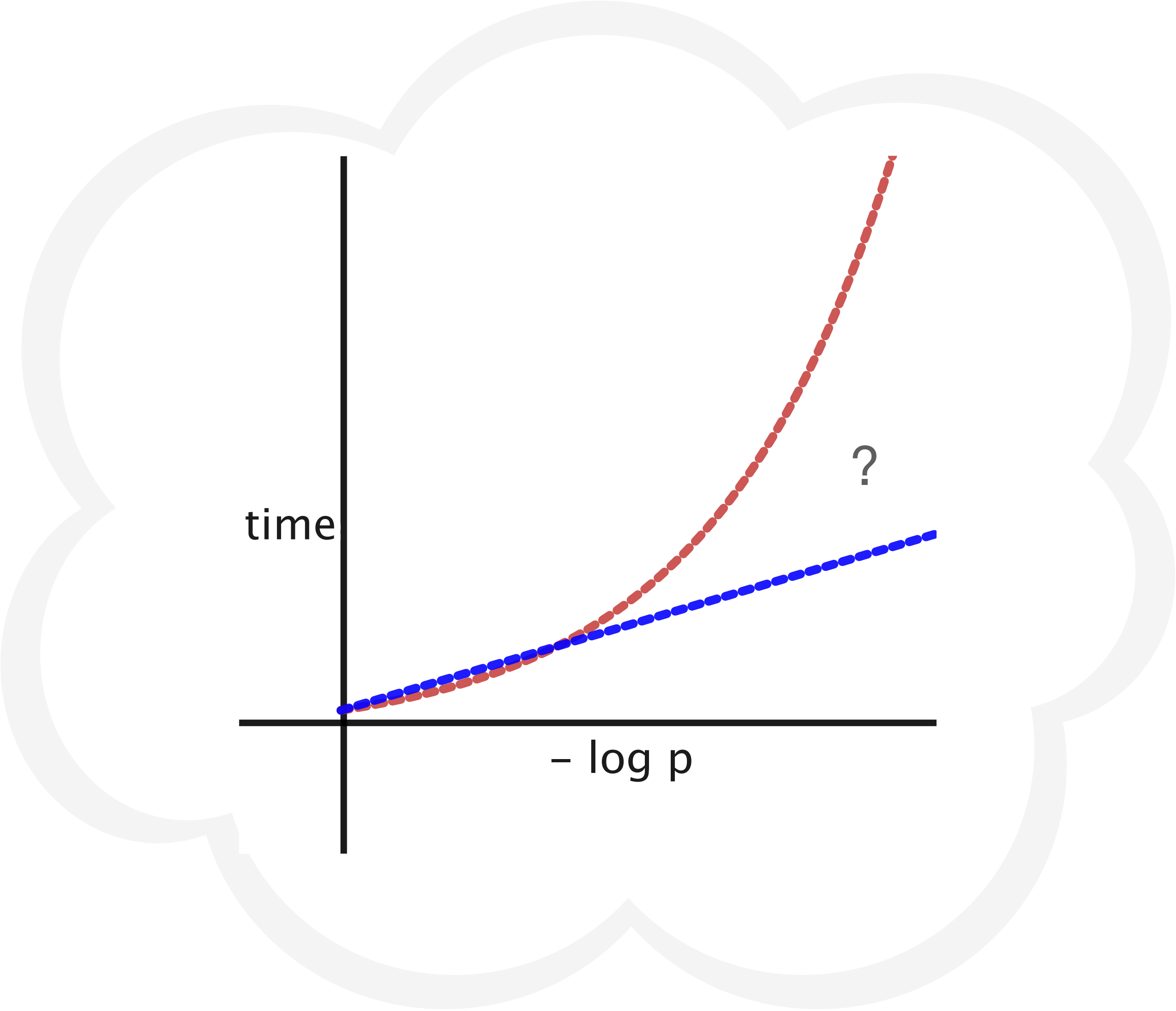
- Earlier papers assume a constant effect of surprisal on variance.

    Hofmann et al. 2022

Our Study
Details

- Want to model relationship between surprisal and reading times.
  - Arbitrary functional shapes.
  - Model arbitrary relationship with variance as well.

→ GAMs (scale-location models; Wood et al. 2016).



Our Study
Details

- Psychometric corpus:
  - High surprisal items (to see superlinear effects).
  - Large number of participants (to control for participant-wise variation).

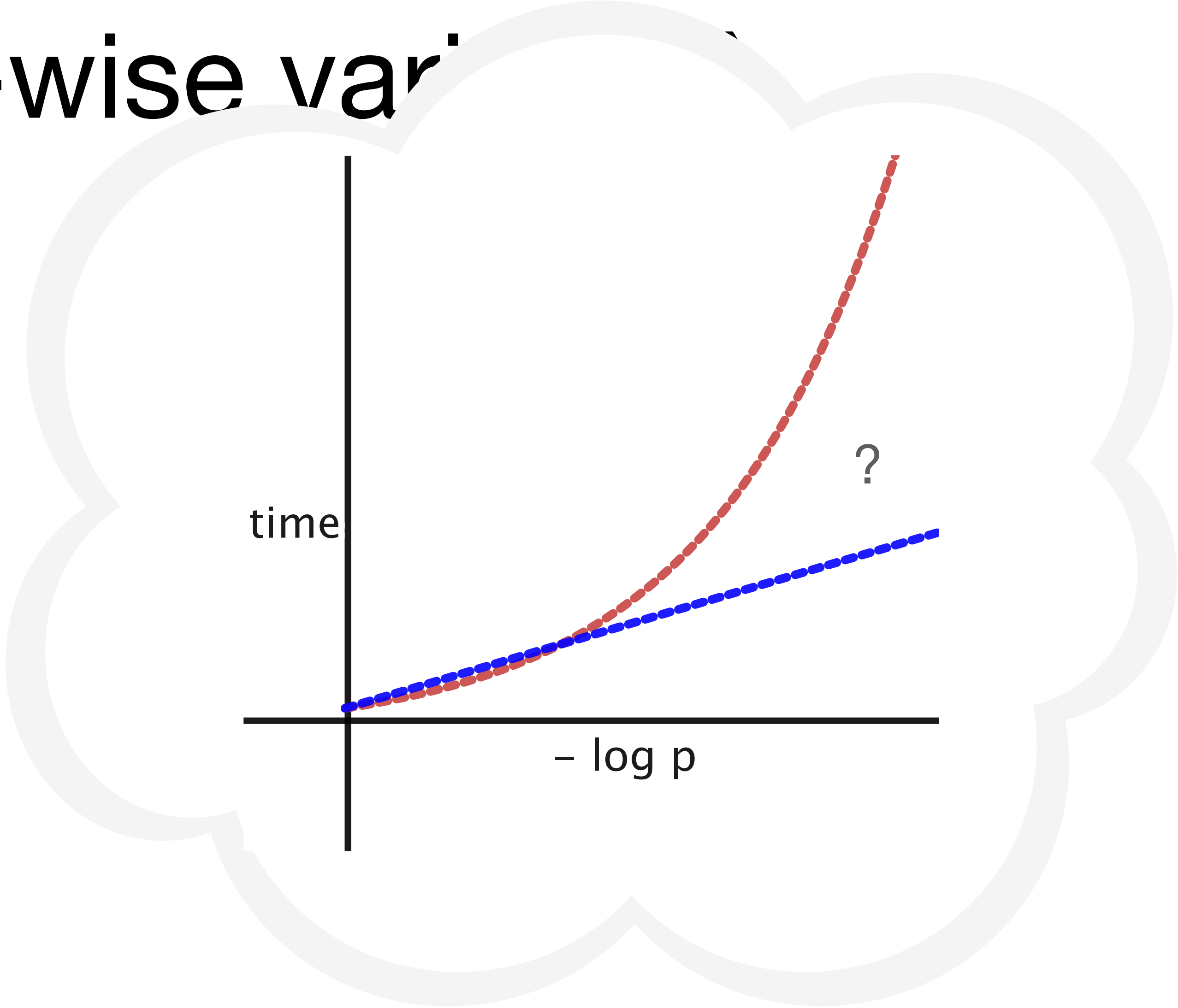
→ Natural Stories (Futrell et al. 2021).

Our Study

Details

- Psychometric corpus:
  - High surprisal items (to see superlinear effects).
  - Large number of participants (to control for participant-wise var

→ Natural Stories (Futrell et al. 2021).



Our Study
Details

- Psychometric corpus:
  - High surprisal items (to see superlinear effects).
  - Large number of participants (to control for participant-wise variation).

→ Natural Stories (Futrell et al. 2021).

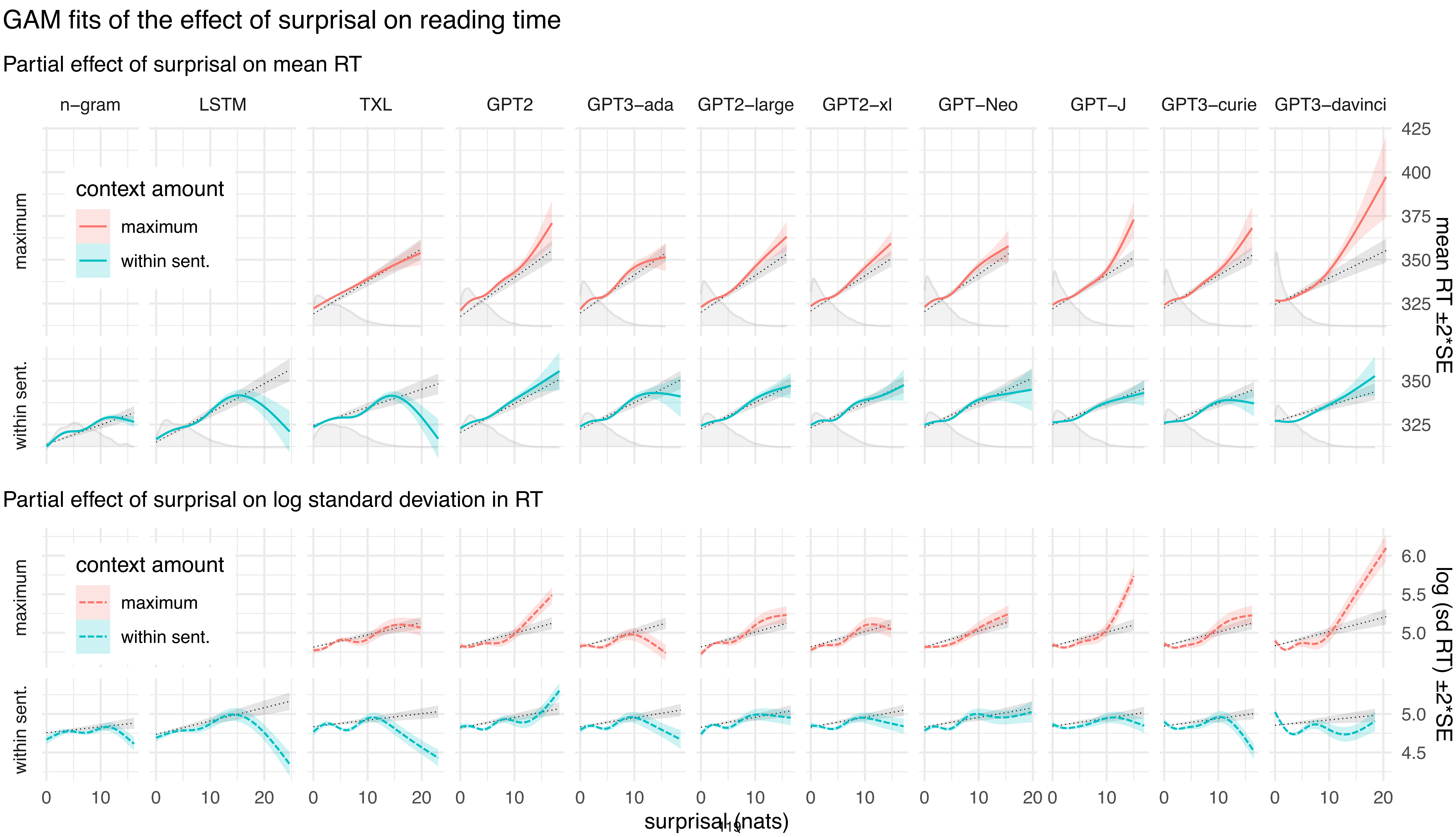
Our Study

Details

- Surprisal estimates from different language models.
  - Most accurate predictions for surprisals.

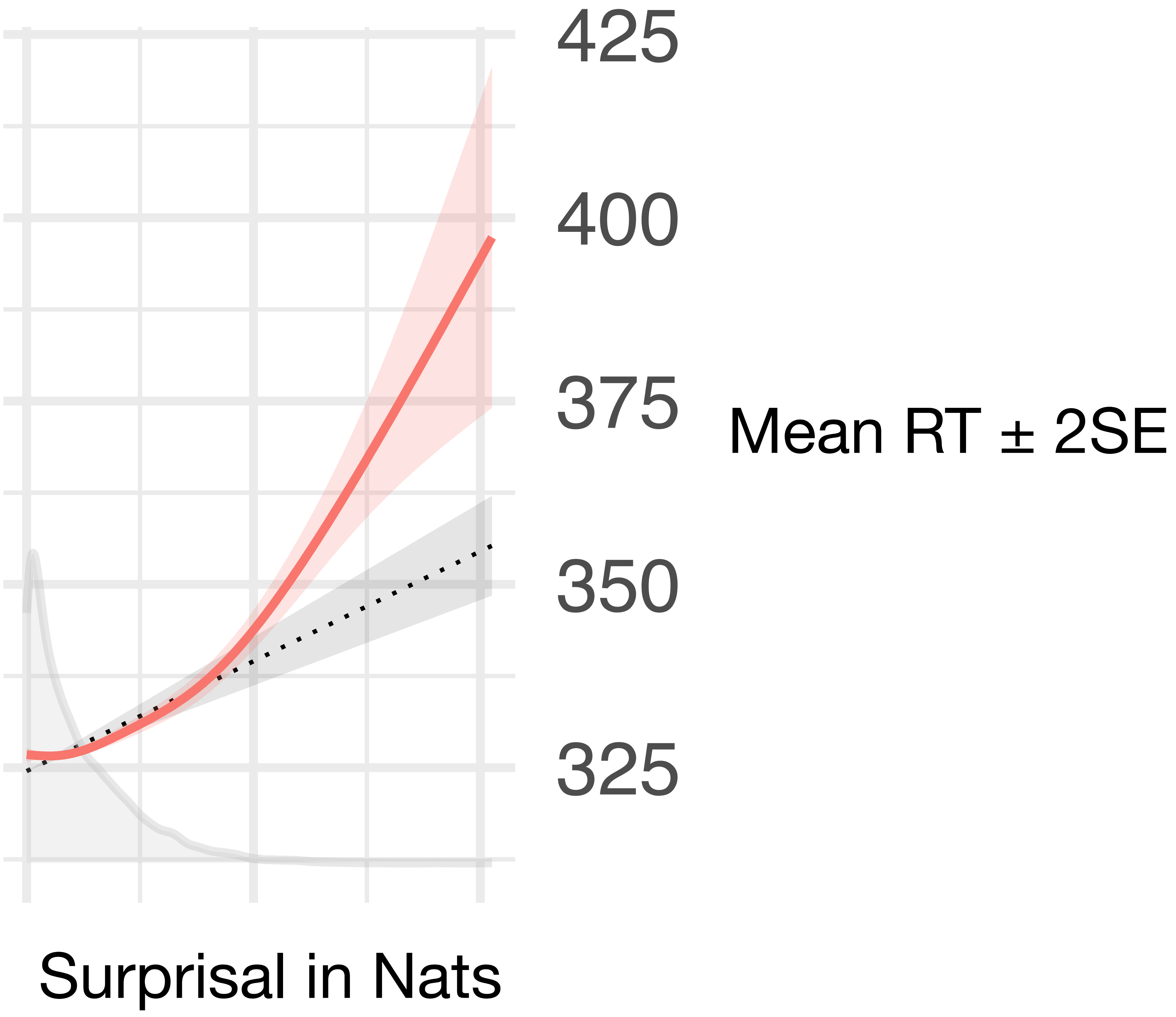
  → Transformer-based LMs (including GPT-3, Brown et al. 2020).

  → Vary amount of context.

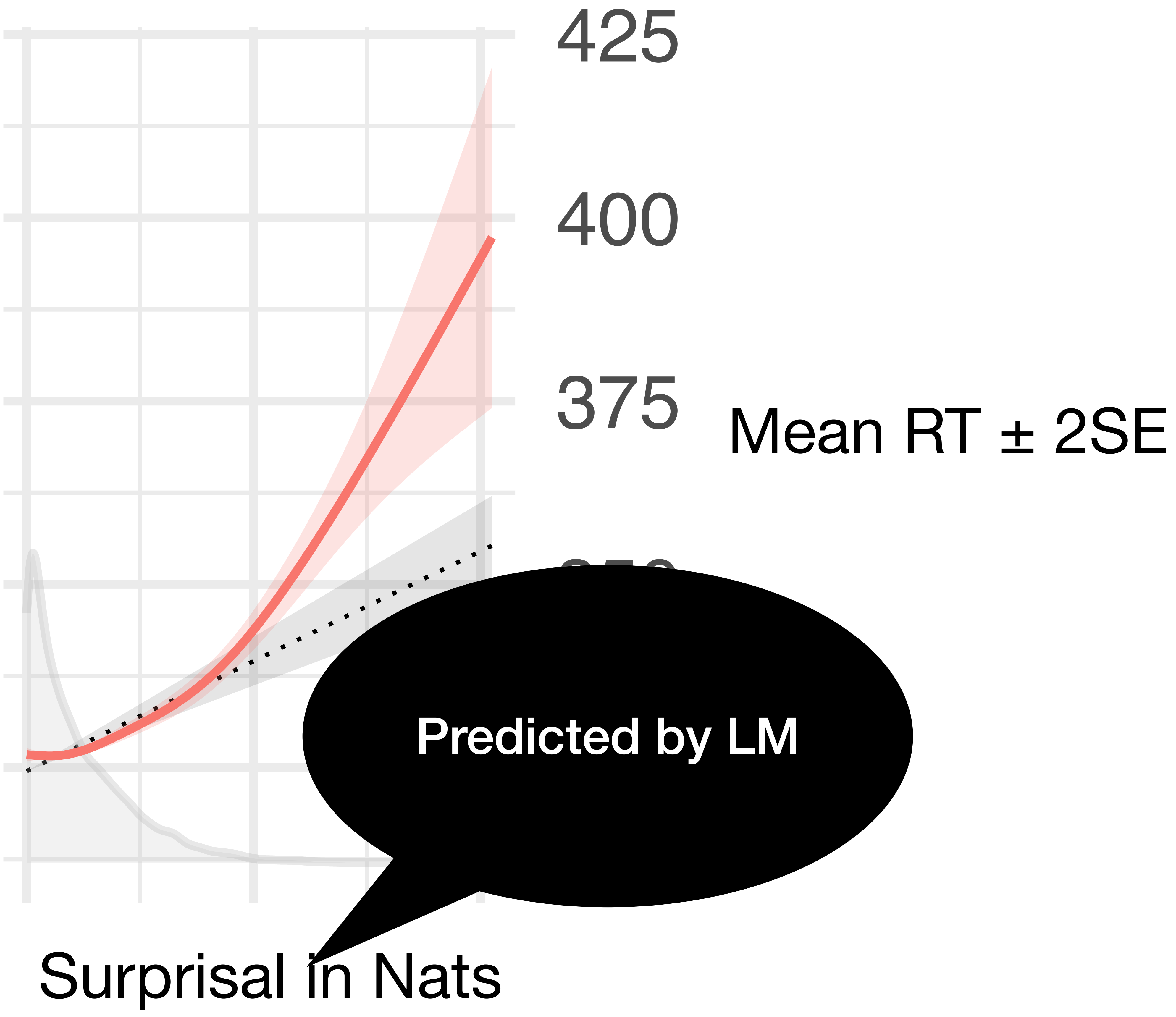


GAM fits of the effect of surprisal on reading time
Partial effect of surprisal on mean RT
n−gram
LSTM
TXL
GPT2
GPT3−ada
GPT2−large
GPT2−xl
GPT−Neo
GPT−J
GPT3−curie
GPT3−davinci
maximum
within sent.
context amount
maximum
within sent.
mean RT ±2*SE
425
400
375
350
325
Partial effect of surprisal on log standard deviation in RT
log (sd RT) ±2*SE
6.0
5.5
5.0
4.5
0
10
20
surprisal (nats)

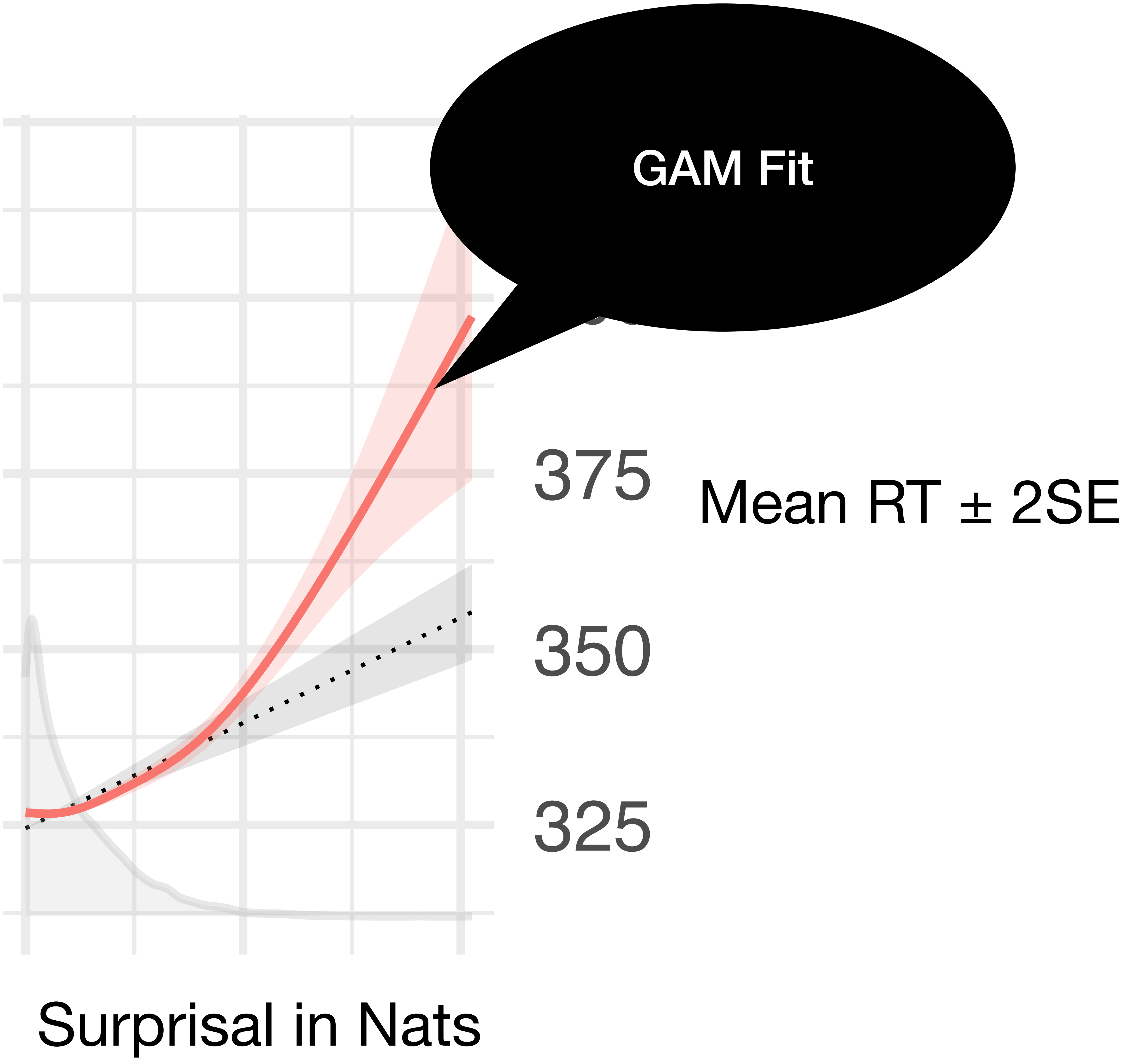
Individual Plots



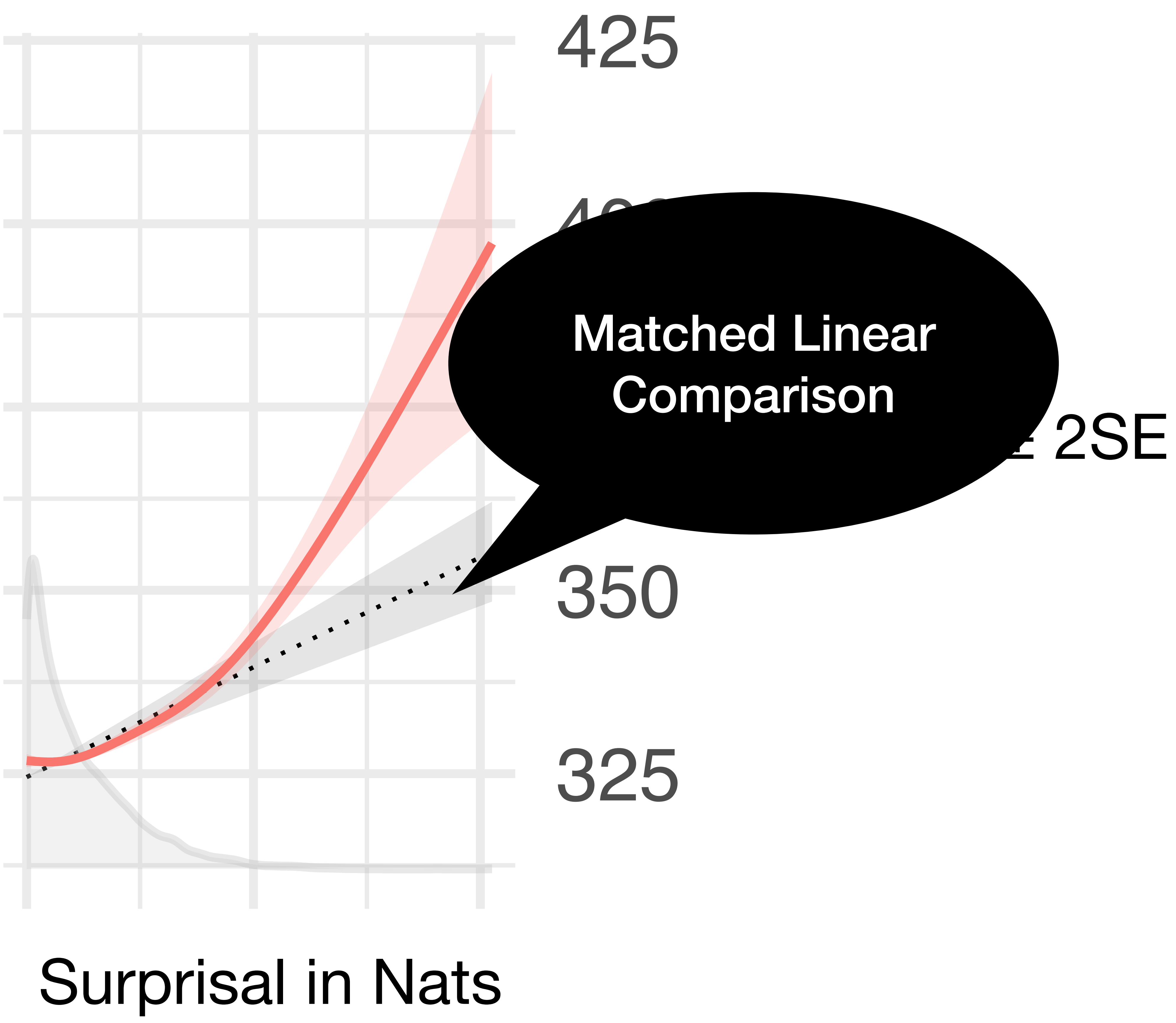
Individual Plots



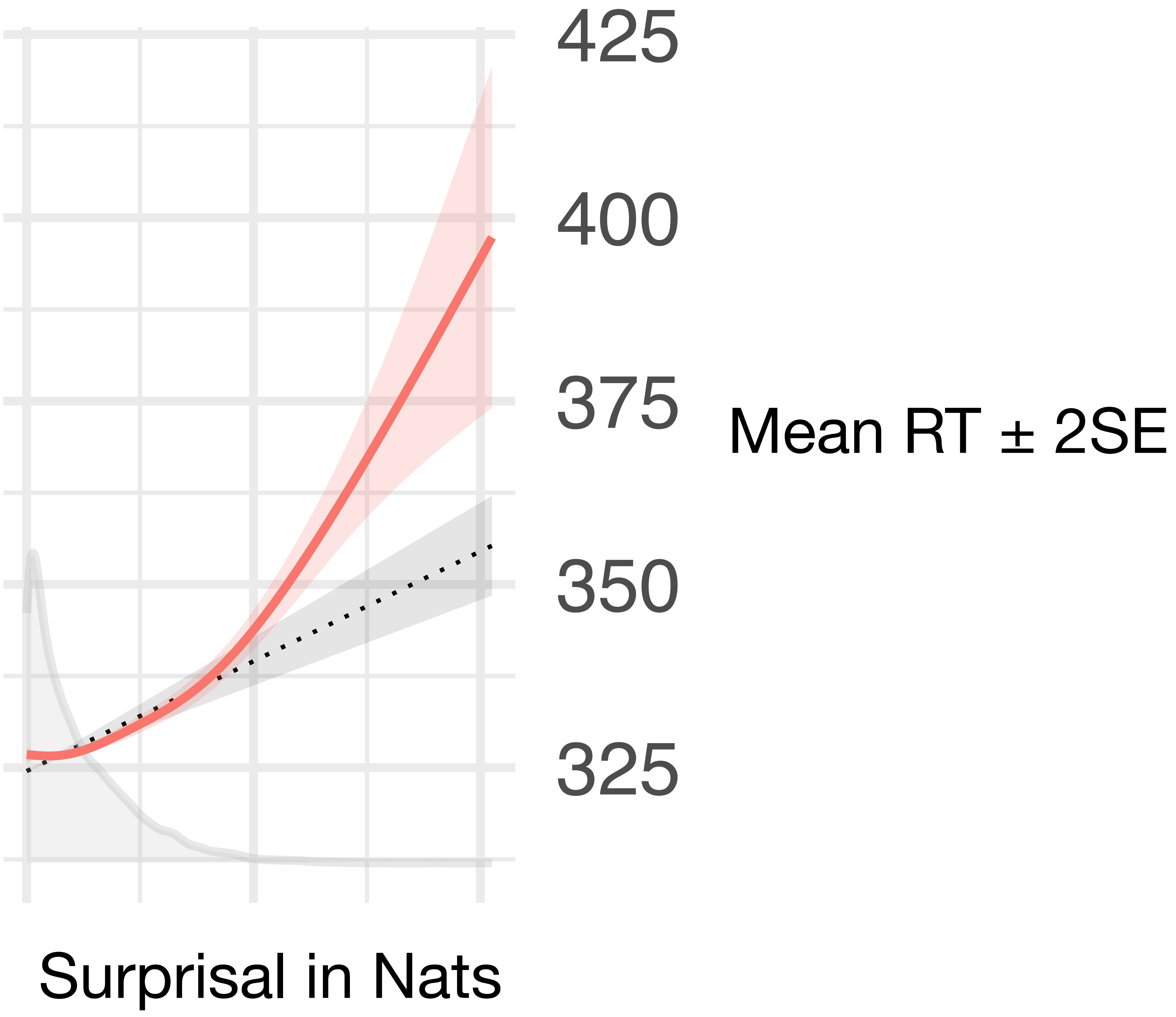
Plots



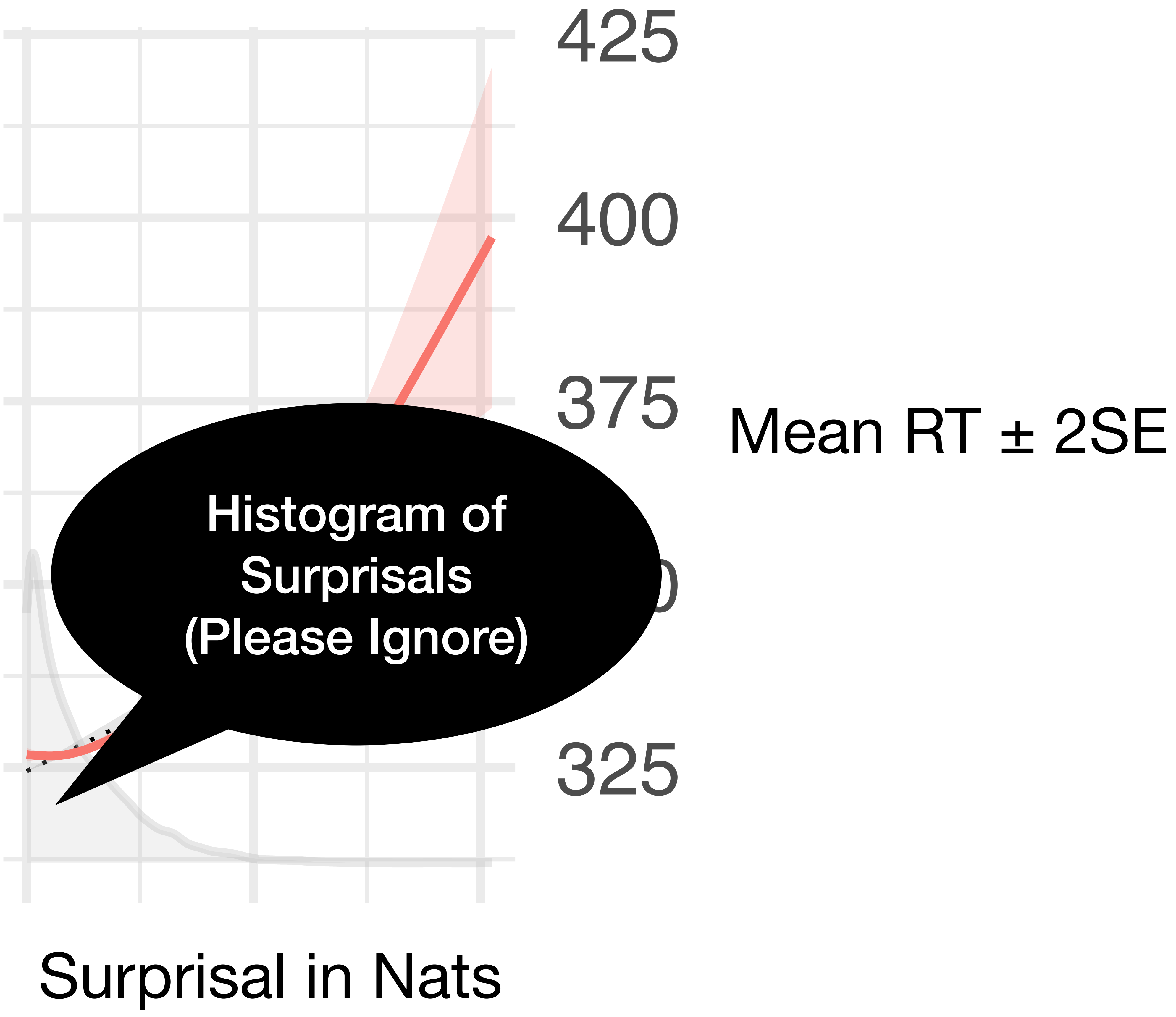
Plots

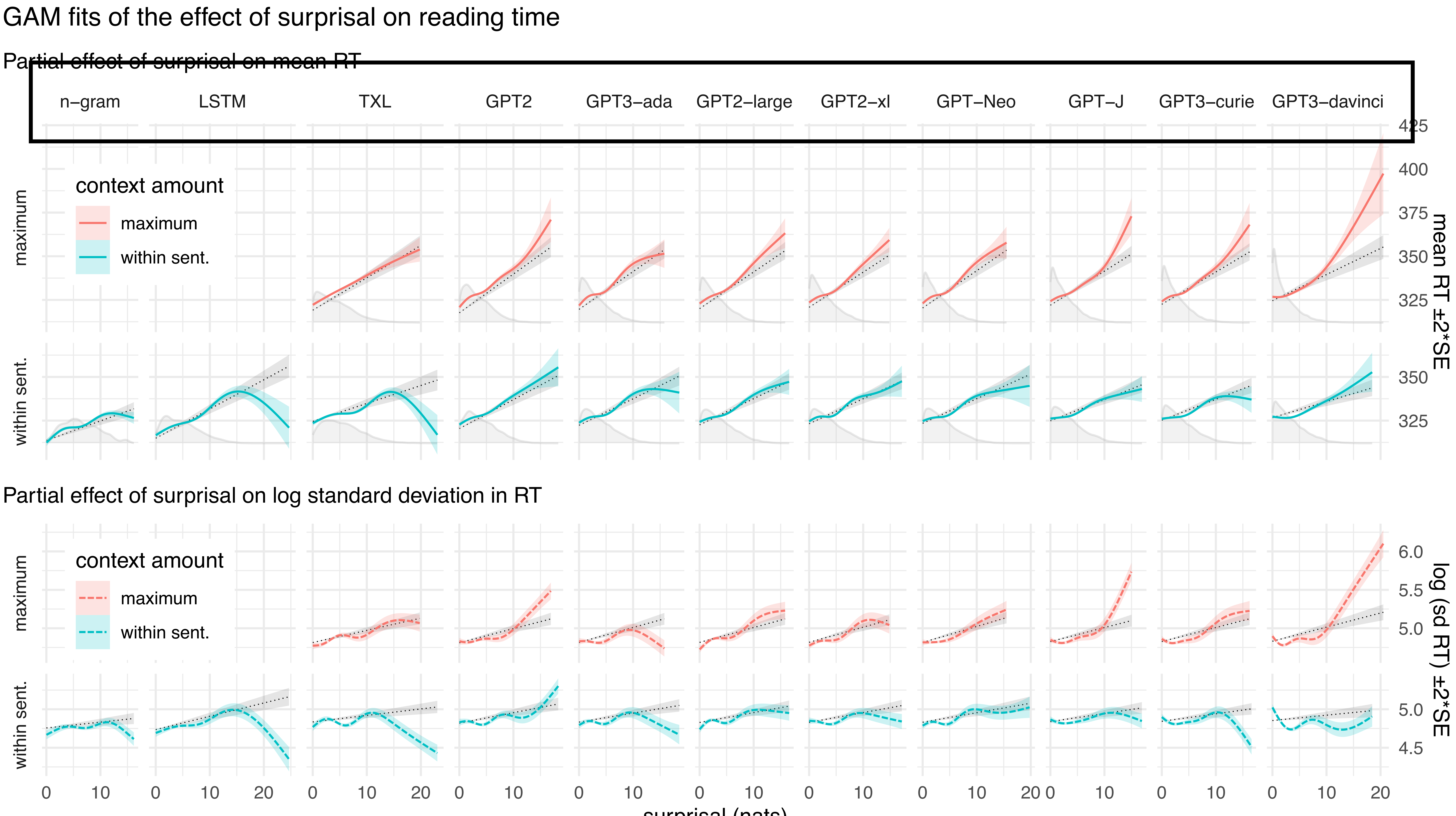

425
350
325
Matched Linear
Comparison
2SE
Surprisal in Nats

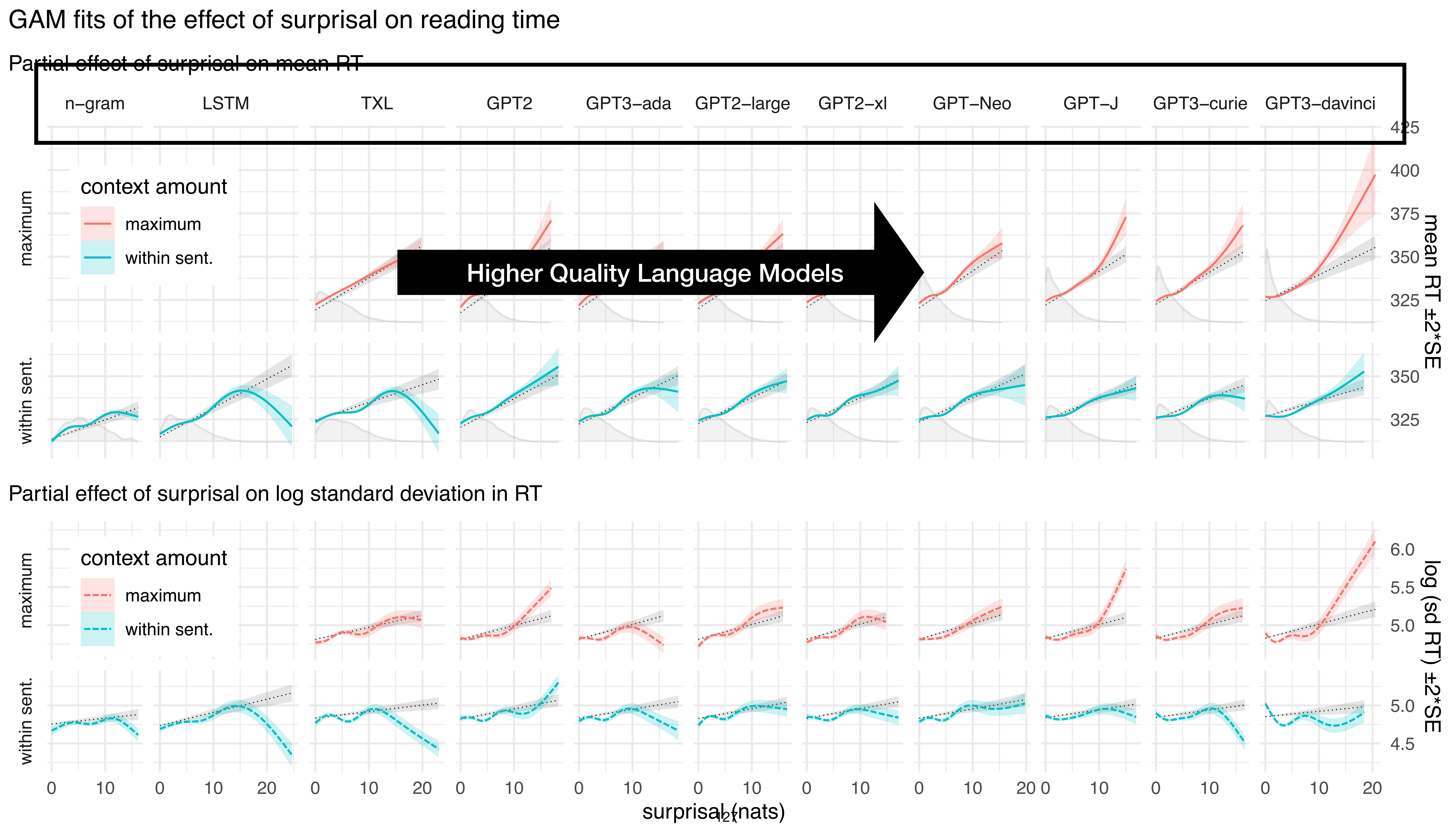
Plots



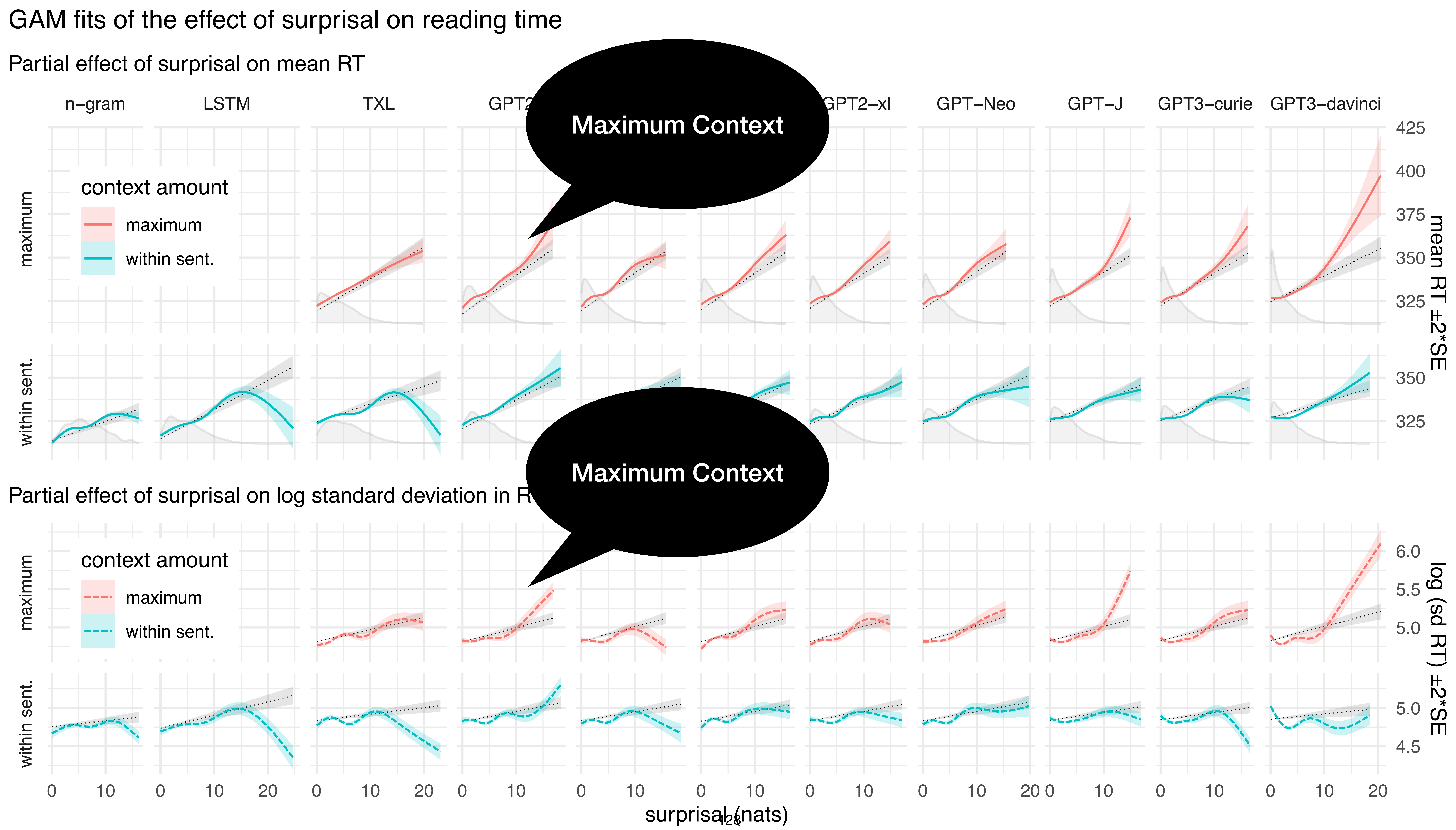
Plots

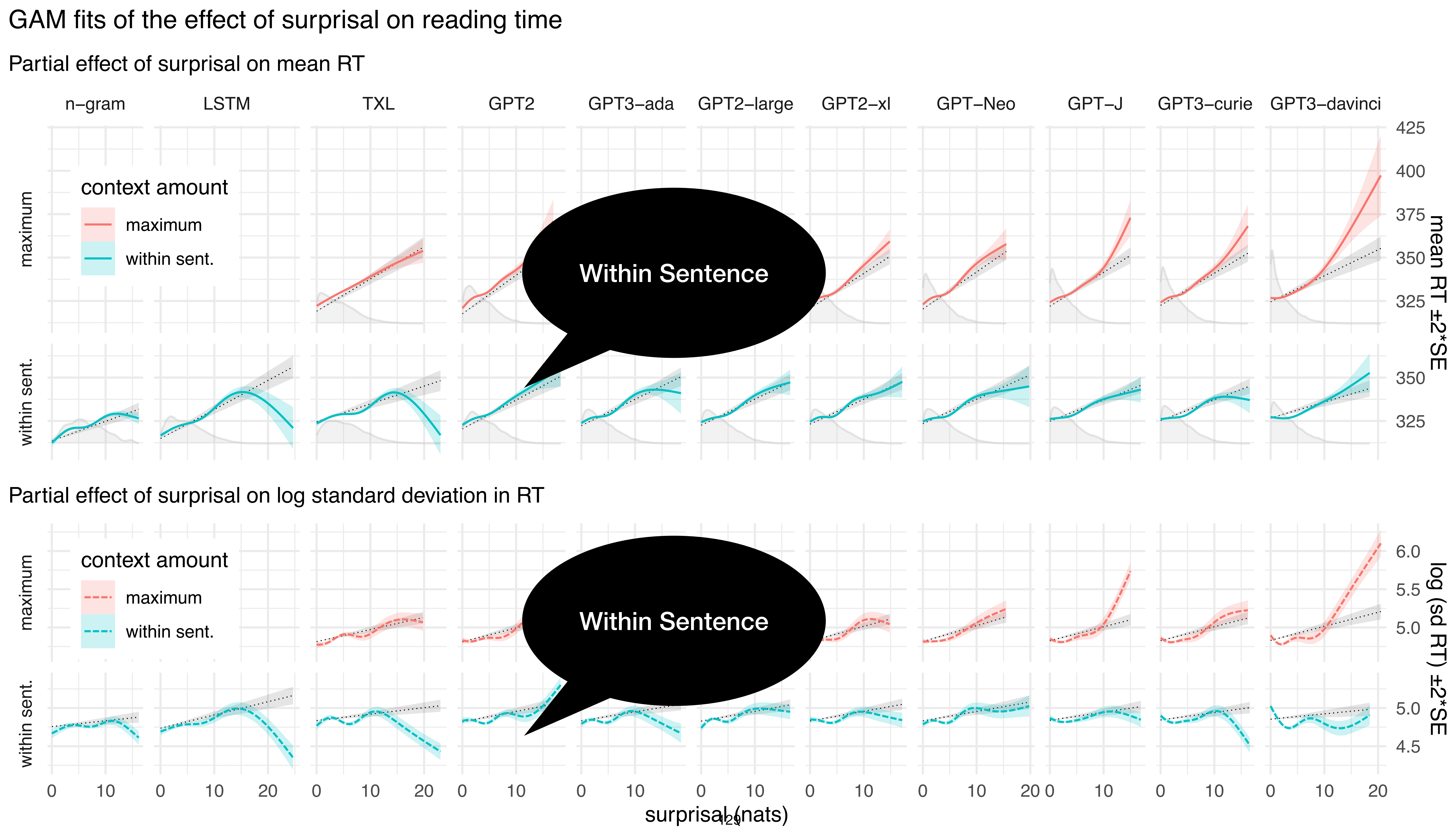

425
400
375
Mean RT ± 2SE
325
Histogram of Surprisals (Please Ignore)
Surprisal in Nats

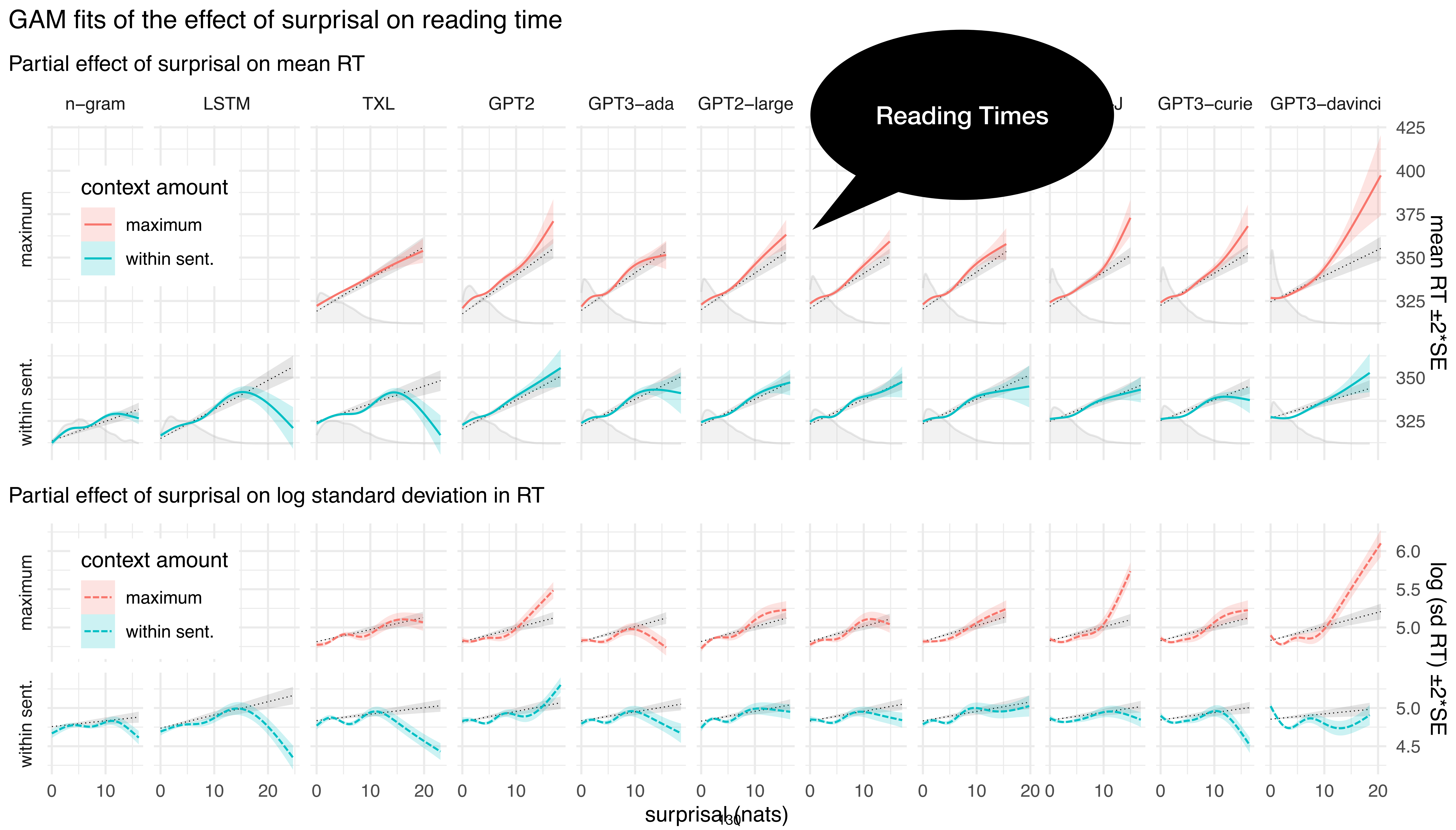

GAM fits of the effect of surprisal on reading time
Partial effect of surprisal on mean RT
n−gram
LSTM
TXL
GPT2
GPT3−ada
GPT2−large
GPT2−xl
GPT−Neo
GPT−J
GPT3−curie
GPT3−davinci
context amount
maximum
within sent.
mean RT ±2*SE
425
400
375
350
325
Partial effect of surprisal on log standard deviation in RT
log (sd RT) ±2*SE
6.0
5.5
5.0
4.5
0
10
20
surprisal (nats)

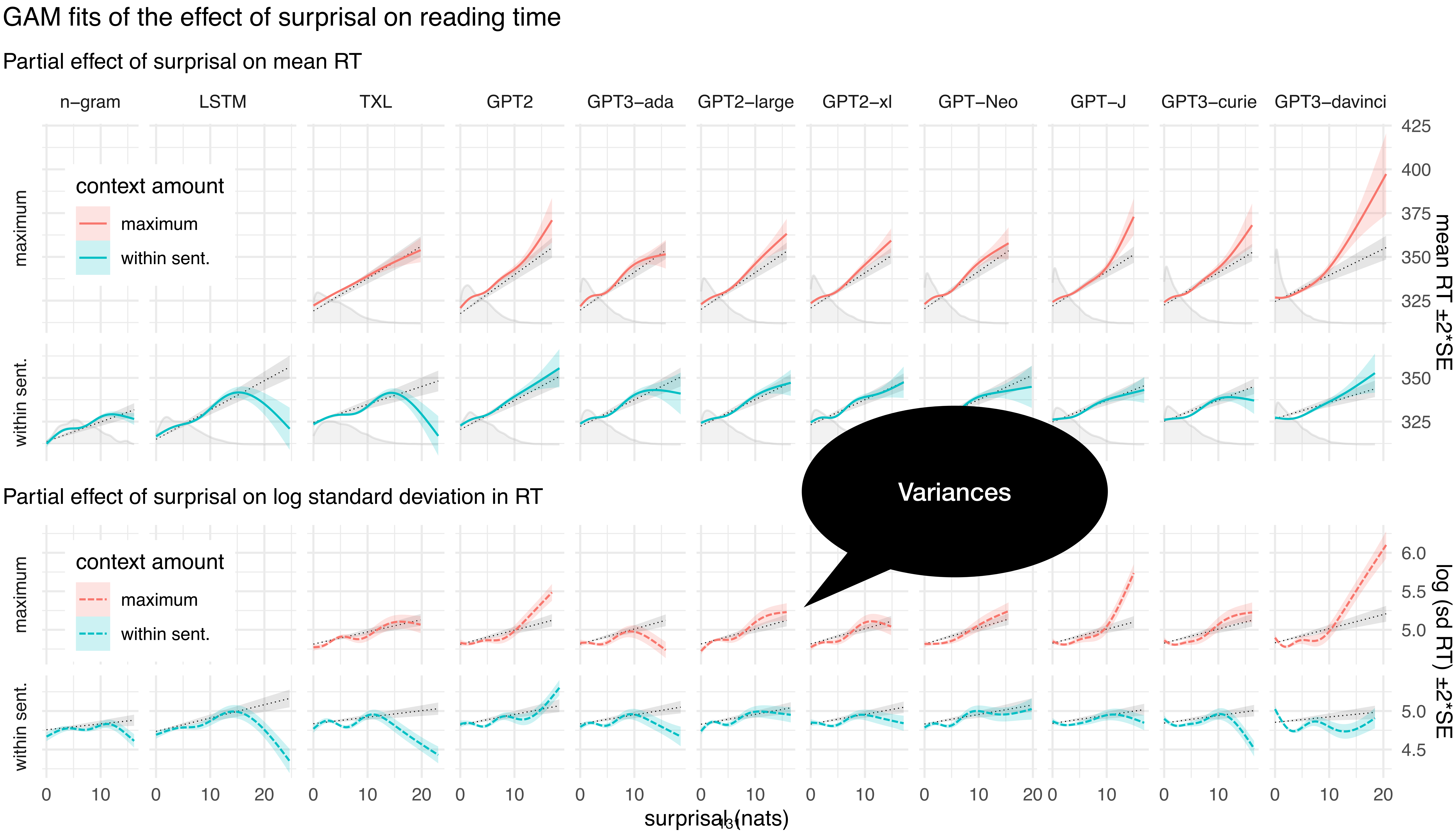


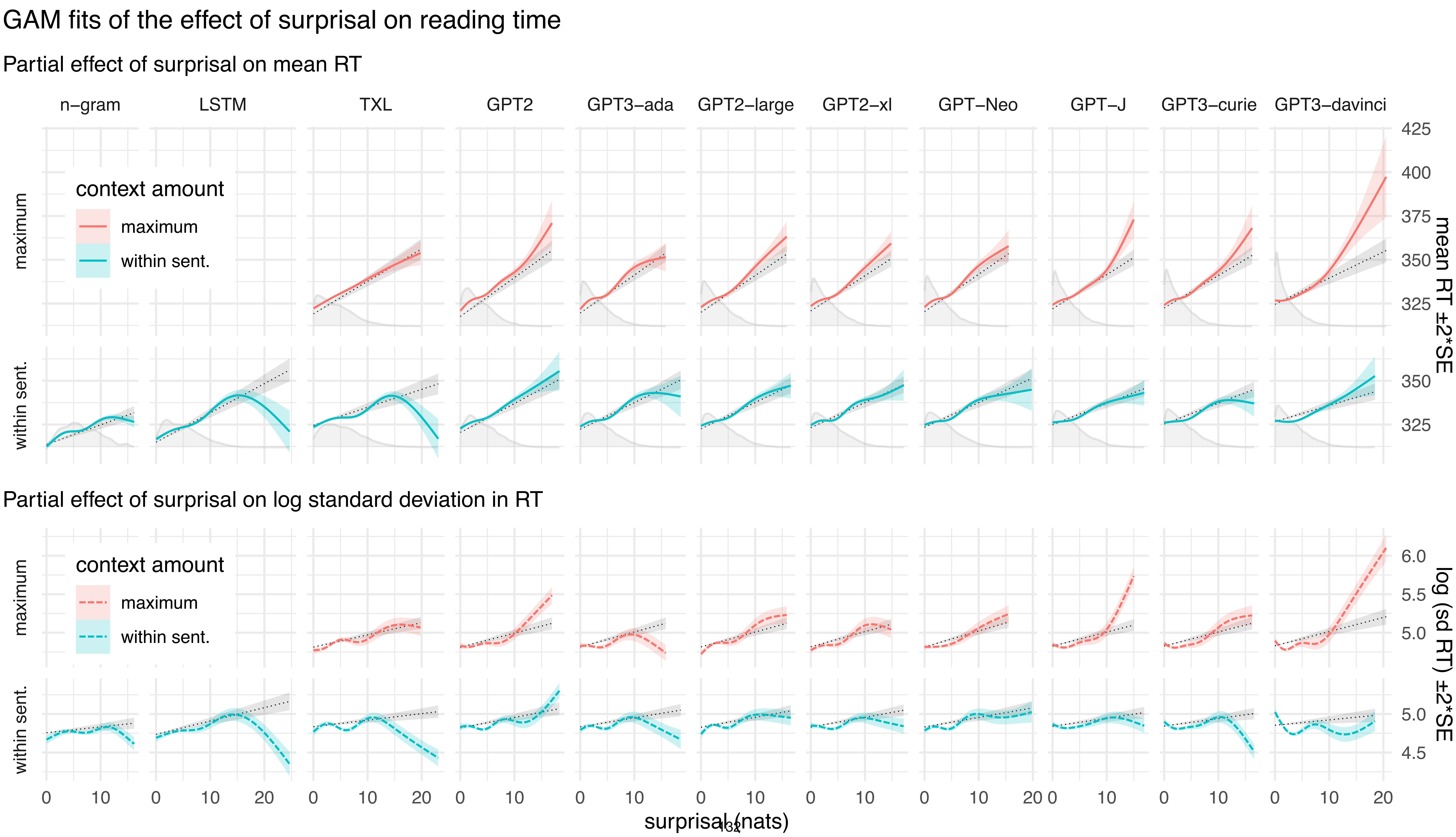
GAM fits of the effect of surprisal on reading time
Partial effect of surprisal on mean RT
n−gram
LSTM
TXL
GPT2
GPT3−ada
GPT2−large
GPT2−xl
GPT−Neo
GPT−J
GPT3−curie
GPT3−davinci
context amount
maximum
within sent.
Higher Quality Language Models
mean RT ±2*SE
Partial effect of surprisal on log standard deviation in RT
log (sd RT) ±2*SE
surprisal (nats)


GAM fits of the effect of surprisal on reading time
Partial effect of surprisal on mean RT
n−gram
LSTM
TXL
GPT2
GPT2−xl
GPT−Neo
GPT−J
GPT3−curie
GPT3−davinci
Maximum Context
context amount
maximum
within sent.
maximum
within sent.
mean RT ±2*SE
425
400
375
350
325
350
325
Maximum Context
Partial effect of surprisal on log standard deviation in R
context amount
maximum
within sent.
maximum
within sent.
log (sd RT) ±2*SE
6.0
5.5
5.0
5.0
4.5
0
10
20
surprisal (nats)


GAM fits of the effect of surprisal on reading time
Partial effect of surprisal on mean RT
n−gram
LSTM
TXL
GPT2
GPT3−ada
GPT2−large
GPT2−xl
GPT−Neo
GPT−J
GPT3−curie
GPT3−davinci
maximum
within sent.
context amount
maximum
within sent.
Within Sentence
mean RT ±2*SE
425
400
375
350
325
350
325
Partial effect of surprisal on log standard deviation in RT
maximum
within sent.
context amount
maximum
within sent.
Within Sentence
log (sd RT) ±2*SE
6.0
5.5
5.0
5.0
4.5
0
10
20
surprisal (nats)


GAM fits of the effect of surprisal on reading time
Partial effect of surprisal on mean RT
n−gram
LSTM
TXL
GPT2
GPT3−ada
GPT2−large
Reading Times
GPT3−curie
GPT3−davinci
context amount
maximum
within sent.
maximum
within sent.
mean RT ±2*SE
425
400
375
350
325
350
325
Partial effect of surprisal on log standard deviation in RT
context amount
maximum
within sent.
maximum
within sent.
log (sd RT) ±2*SE
6.0
5.5
5.0
5.0
4.5
0
10
20
surprisal (nats)


GAM fits of the effect of surprisal on reading time
Partial effect of surprisal on mean RT
n−gram
LSTM
TXL
GPT2
GPT3−ada
GPT2−large
GPT2−xl
GPT−Neo
GPT−J
GPT3−curie
GPT3−davinci
maximum
within sent.
context amount
maximum
within sent.
425
400
375
350
325
350
325
mean RT ±2*SE
Variances
Partial effect of surprisal on log standard deviation in RT
context amount
maximum
within sent.
6.0
5.5
5.0
5.0
4.5
log (sd RT) ±2*SE
0
10
20
surprisal (nats)


GAM fits of the effect of surprisal on reading time
Partial effect of surprisal on mean RT
n−gram
LSTM
TXL
GPT2
GPT3−ada
GPT2−large
GPT2−xl
GPT−Neo
GPT−J
GPT3−curie
GPT3−davinci
maximum
within sent.
context amount
maximum
within sent.
mean RT ±2*SE
425
400
375
350
325
Partial effect of surprisal on log standard deviation in RT
log (sd RT) ±2*SE
6.0
5.5
5.0
4.5
0
10
20
surprisal (nats)

Interpretation

- Evidence for a non-linear effect of surprisal on processing times.
  - The better the LM (and more context) the larger the effect.
  - May be why earlier studies failed to find such an effect.
- Evidence for an increase in variance with surprisal at least in best LMs.
  - Evidence against probability ordered sequential search.

Compositionality and Incremental Processing

- Presented a modeling framework that can capture compositionality and incrementality in human sentence processing.
  - Early prototype.
- Considered the sequential inference problem associated with this framework, and sequential importance sampling as a possible solution.
- Raised an important potential problem with sampling as a model of humans: inconsistency with surprisal theory.
- Showed that perhaps human scaling is in fact superlinear.

Thanks!