Supplementary Information for: Crosslinguistic Word Orders Enable an Efficient Tradeoff between Memory and Surprisal

Michael Hahn, Judith Degen, Richard Futrell

2018

Contents

1 Formal Analysis and Proofs

In this section, we prove Theorem 1.

1.1 Mathematical Assumptions

We first make explicit how we formalize language processing for proving the theorem.

Ingredient 1: Language as a Stationary Stochastic Process We represent language as a stochastic process of words $\dots w_{-2}w_{-1}w_0w_1w_2\dots$, extending indefinitely both into the past and into the future. The symbols w_i belong to a common set, representing the words of the language.[1]

The assumption of infinite length is for mathematical convenience and does not affect the substance of our results: As we restrict our attention to the processing of individual sentences, which have finite length, we will actually not make use of long-range and infinite contexts.

We make the assumption that this process is *stationary*. Formally, this means that the conditional distribution $P(w_t|w_{<t})$ does not depend on t, it only depends on the (semi-infinite) context sequence $w_{<t}$. Informally, this says that the process has no 'internal clock', and that the statistical rules of the language do not change at the timescale we are interested in. In reality, the statistical rules of language do change: They change as language changes over generations, and they also change between different situations – e.g., depending on the interlocutor at a given point in time. However, we are interested in memory needs in the processing of *individual sentences*, at a timescale of seconds or minutes. At this level, the statistical regularities of language do not change, making stationarity a reasonable modeling assumption.

Ingredient 2: Postulates about Processing The second ingredient consists of the three postulates described in the main paper. There are no further assumptions about the memory architecture and the nature of its computations.

[1] Could also be phonemes, sentences, or any other kind of unit.

1.2 Proof of the Theorem

We restate the theorem:

Theorem 1. *Let T be any positive integer ($T \in \{1,2,3,...\}$), and consider a listener using at most*

$$\sum_{t=1}^{T} tI_t \tag{1}$$

bits of memory on average. Then this listener will incur surprisal at least

$$H[w_t|w_{<t}] + \sum_{t>T} I_t$$

on average.

Proof. The difference between the listener's average surprisal S_M and optimal surprisal S_∞ is $S_M - S_\infty = \mathrm{H}[w_t|m_t] - \mathrm{H}[w_t|w_{<t}]$.[2] By the assumption of stationarity, we can, for any positive integer T, rewrite this expression as

$$\mathrm{H}[w_t|m_t] - \mathrm{H}[w_t|w_{<t}] = \frac{1}{T}\sum_{t'=1}^{T}\left(\mathrm{H}[w_{t'}|m_{t'}] - \mathrm{H}[w_{t'}|w_{<t'}]\right) \tag{2}$$

Because m_t is determined by $(w_{1\ldots t-1}, m_1)$:

$$m_t = M(m_{t-1}, w_{t-1}) = M(M(m_{t-2}, w_{t-2}), w_{t-1}) = M(M(M(m_{t-3}, w_{t-3}), w_{t-2}), w_{t-1}) = \ldots \tag{3}$$

the Data Processing inequality entails the following inequality for every positive integer t:

$$\mathrm{H}[w_t|m_t] \geq \mathrm{H}[w_t|w_{1\ldots t-1}, m_1] \tag{4}$$

Plugging this inequality into Equation **??** above:

$$\mathrm{H}[w_t|m_t] - \mathrm{H}[w_t|w_{<t}] \geq \frac{1}{T}\sum_{t=1}^{T}(\mathrm{H}[w_t|w_{1\ldots t-1}, m_1] - \mathrm{H}[w_t|w_{1\ldots t-1}, w_{\leq 0}]) \tag{5}$$

$$= \frac{1}{T}\left(\mathrm{H}[w_{1\ldots T}|m_1] - \mathrm{H}[w_{1\ldots T}|w_{\leq 0}]\right) \tag{6}$$

$$= \frac{1}{T}\left(I[w_{1\ldots T}, w_{\leq 0}] - I[w_{1\ldots T}, m_1]\right) \tag{7}$$

The first term $I[w_{1\ldots T}, w_{\leq 0}]$ can be rewritten in terms of I_t:

$$I[w_{1\ldots T}, w_{\leq 0}] = \sum_{i=1}^{T}\sum_{j=-1}^{-\infty} I[w_i, w_j|w_{j+1}\ldots w_{i-1}] = \sum_{t=1}^{T} tI_t + T\sum_{t>T} I_t \tag{8}$$

Therefore

$$\mathrm{H}[w_t|m_t] - \mathrm{H}[w_t|w_{<t}] \geq \frac{1}{T}\left(\sum_{t=1}^{T} tI_t + T\sum_{t>T} I_t - I[w_{1\ldots T}, m_1]\right) \tag{9}$$

[2] A listener whose predictions are not optimal given m_t can only incur even higher surprisal.

The term $I[w_{1...T}|m_1]$ is at most $\mathrm{H}[m_1]$, which is at most $\sum_{t=1}^{T} tI_t$ by assumption. Thus, (**??**) implies the following:

$$\begin{aligned}\mathrm{H}[w_t|m_t] - \mathrm{H}[w_t|w_{<t}] &\geq \frac{1}{T}\left(\sum_{t=1}^{T} tI_t + T\sum_{t>T} I_t - \sum_{t=1}^{T} tI_t\right) \\ &= \sum_{t>T} I_t\end{aligned}$$

Rearranging yields

$$\mathrm{H}[w_t|m_t] \geq \mathrm{H}[w_t|w_{<t}] + \sum_{t>T} I_t \tag{10}$$

as claimed. □

1.3 For nondeterministic encoding functions

We have been assuming that m_t is a deterministic function of x_t and m_{t-1}. Here, we show that this assumption can be relaxed to stochastic encoding functions. That is, we show that our result holds if m_t is not a deterministic function of w_{t-1} and m_{t-1}, but includes randomness in processing.

To formalize that setting, we relax Comprehension Postulate 1 to the following requirement, for all values of $m_1, (w_t)_{t\in\mathbb{Z}}, m_0$:

$$p(m_1|(w_t)_{t\in\mathbb{Z}}, m_0) = p(m_1|m_0, w_1) \tag{11}$$

This says that m_1 contains no information about the utterances beyond what is contained in m_0 and w_1.

The one place in the proof where Comprehension Postulate 1 plays a role is the proof of the inequality:

$$H[w_t|m_t] \geq H[w_t|w_{1...t-1}, m_1] \tag{12}$$

We show that this inequality still holds under the relaxed condition (**??**):

Proof. By Bayes' Theorem

$$\begin{aligned}p(w_t|m_0, m_1, w_{0...t-1}) &= \frac{p(m_1|m_0, w_{0...t})}{p(m_1|m_0, w_{0...t-1})} \cdot p(w_t|m_0, w_{0...t-1}) \\ &= \frac{p(m_1|m_0, w_0)}{p(m_1|m_0, w_0)} \cdot p(w_t|m_0, w_{0...t-1}) \\ &= p(w_t|m_0, w_{0...t-1})\end{aligned}$$

where the second equation follows from (**??**). So we have a Markov chain

$$(w_t) \to (m_0, w_{0...t-1}) \to (m_1, w_{1...t-1}) \tag{13}$$

Thus, by the Data Processing Inequality,

$$H[w_t|w_{1...t-1}, m_1] \geq H[w_t|w_{0...t-1}, m_0] \tag{14}$$

Finally, iteratively applying this reasoning, we conclude:

$$H[w_t|m_t] \geq H[w_t|w_{t-1}, m_{t-1}] \geq H[w_t|w_{t-2,t-1}, m_{t-2}] \geq \ldots \geq H[w_t|w_{1...t-1}, m_1]$$

□

1.4 Locality in a model with Memory Retrieval

Here we show that our information-theoretic analysis is compatible with models placing the main bottleneck in the difficulty of retrieval (**????**). We extend our model of memory in incremental prediction to capture key aspects of the models described by **???**.

The ACT-R model of **?** assumes a small working memory consisting of *buffers* and a *control state*, which together hold a small and fixed number of individual *chunks*. It also assumes a large short-term memory that contains an unbounded number of chunks. This large memory store is accessed via *cue-based retrieval*: a query is constructed based on the current state of the buffers and the control state; a chunk that matches this query is then selected from the memory storage and placed into one of the buffers.

Formal Model We extend our information-theoretic analysis by considering a model that maintains both a small working memory m_t – corresponding to the buffers and the control state – and an unlimited short-term memory s_t. Predictions are made based on working memory m_t, incurring surprisal $H[w_t|m_t]$. When processing a word x_t, there is some amount of communication between m_t and s_t, corresponding to retrieval operations. We model this using a variable r_t representing the information that is retrieved from s_t. In our formalization, r_t reflects the totality of all retrieval operations that are made during the processing of x_{t-1}; they happen after x_{t-1} has been observed but before x_t has.

The working memory state is determined not just by the input x_t and the previous working memory state m_{t-1}, but also by the retrieved information:

$$m_t = f(x_t, m_{t-1}, r_t) \tag{15}$$

The retrieval operation is jointly determined by working memory, short-term memory, and the previous word:

$$r_t = g(x_{t-1}, m_{t-1}, s_{t-1}) \tag{16}$$

Finally, the short-term memory can incorporate any – possibly all – information from the last word and the working memory:

$$s_t = h(x_{t-1}, m_{t-1}, s_{t-1}) \tag{17}$$

While s_t is unconstrained, there are constraints on the capacity of working memory $\mathrm{H}[m_t]$ and the amount of retrieved information $\mathrm{H}[r_t]$. Placing a bound on $\mathrm{H}[m_t]$ reflects the fact that the buffers can only hold a small and fixed number of chunks (**?**).

Cost of Retrieval In the model of **?**, the time it takes to process a word is determined primarily by the time spent retrieving chunks, which is determined by the number of retrieval operations and the time it takes to complete each retrieval operation. If the information content of each chunk is bounded, then a bound on $H[r_t]$ corresponds to a bound on the number of retrieval operations.

In the model of **?**, a retrieval operation takes longer if more chunks are similar to the retrieval cue, whereas, in the direct-access model (**???**), retrieval operations take a constant amount of time. There is no direct counterpart to differences in retrieval times and similarity-based inhibition as in the activation-based model in our formalization. Our formalization thus more closely matches the direct-access model, though it might be possible to incorporate aspects of the activation-based model in our formalization.

Role of Surprisal The ACT-R model of **?** does not have an explicit surprisal cost. Instead, surprisal effects are interpreted as arising because, in less constraining contexts, the parser is more likely to make decisions that then turn out to be incorrect, leading to additional correcting steps. We view this as an algorithmic-level implementation of a surprisal cost $H[x_t|m_{t-1}]$: If the word x_t is unexpected given the current state of the working memory – i.e., buffers and control states – then their current state must provide insufficient information to constrain the actual syntactic state of the sentence, meaning that the parsing steps made to integrate x_t are likely to include more backtracking and correction steps. Thus, we argue that cue-based retrieval models predict that the surprisal $-\log P(x_t|m_{t-1})$ will be part of the cost of processing word x_t.

Theoretical Result We now show an extension of our theoretical result in the setting of the retrieval-based model described above.

Theorem 2. *Let $0 < S \leq T$ be positive integers such that the average working memory cost $\mathrm{H}[m_t]$ is bounded as*

$$\mathrm{H}[m_t] \leq \sum_{t=1}^{T} tI_t \tag{18}$$

and the average amount of retrieved information is bounded as

$$\mathrm{H}[r_t] \leq \sum_{t=T+1}^{S} I_t \tag{19}$$

Then the surprisal cost is lower-bounded as

$$\mathrm{H}[w_t|m_t] \geq \mathrm{H}[w_t|x_{<t}] + \sum_{t>S} I_t \tag{20}$$

Proof. The proof is a generalization of the proof above. For any positive integer t, m_t is determined by $w_{1\ldots t}, m_0, r_0, \ldots, r_t$. Therefore, the Data Processing Inequality entails:

$$\mathrm{H}[w_t|m_t] \geq \mathrm{H}[w_t|w_{1\ldots t}, m_0, r_0, \ldots, r_t] \tag{21}$$

As in (**??**), this leads to

$$\begin{aligned}
\mathrm{H}[w_t|m_t] - \mathrm{H}[w_t|w_{<t}] &\geq \frac{1}{T}\sum_{t=1}^{T}(\mathrm{H}[w_t|w_{1\ldots t}, m_0, r_0, \ldots, r_t] - \mathrm{H}[w_t|w_{1\ldots t-1}, w_{\leq 0}]) && (22)\\
&\geq \frac{1}{T}\left(\mathrm{H}[w_{1\ldots T}|m_0, r_0, \ldots, r_T] - \mathrm{H}[w_{1\ldots T}|w_{\leq 0}]\right) && (23)\\
&= \frac{1}{T}\left(I[w_{1\ldots T}, w_{\leq 0}] - I[w_{1\ldots T}, (m_0, r_0, \ldots, r_T)]\right) && (24)
\end{aligned}$$

Now, using the calculation from (**??**), this can be rewritten as:

$$\begin{aligned}
\mathrm{H}[w_t|m_t] - \mathrm{H}[w_t|w_{<t}] &= \frac{1}{T}\left(\sum_{t=1}^{T} tI_t + T\sum_{t>T} I_t - I[X_1 \ldots X_T, (M_0, R_1, \ldots, R_T)]\right)\\
&= \frac{1}{T}\left(\sum_{t=1}^{T} tI_t + T\sum_{t>T} I_t - I[X_{1\ldots T}, M_0] - \sum_{t=1}^{T} I[X_{1\ldots T}, R_t|M_0, r_{1\ldots t-1}]\right)
\end{aligned}$$

Due to the inequalities $I[X_{1...T}, M_0] \leq \mathrm{H}[M_0]$ and $\mathrm{I}[X_{1...T}, R_t | M_0, r_{1...t-1}] \leq \mathrm{H}[R_t]$, this can be bounded as

$$\mathrm{H}[w_t|m_t] - \mathrm{H}[w_t|w_{<t}] \geq \frac{1}{T}\left(\sum_{t=1}^{T} tI_t + T\sum_{t>T} I_t - H[M_0] - \sum_{t=1}^{T} H[R_t]\right) \tag{25}$$

$$\tag{26}$$

Finally, this reduces as

$$\begin{aligned}
\mathrm{H}[w_t|m_t] - \mathrm{H}[w_t|w_{<t}] &\geq \frac{1}{T}(T\sum_{t>T} I_t - T\cdot H[R_t]) && (27)\\
&= \sum_{t>T} I_t - H[R_t] && (28)\\
&\geq \sum_{t>T} I_t - \sum_{t=T+1}^{S} I_t && (29)\\
&= \sum_{t>S} I_t && (30)
\end{aligned}$$

□

Information Locality We now show that this result predicts information locality provided that retrieving information is more expensive than keeping the same amount of information in working memory. For this, we formalize the problem of finding an optimal memory strategy as a multi-objective optimization, aiming to minimize

$$\lambda_1 H[m_t] + \lambda_2 H[r_t] \tag{31}$$

to achieve a given surprisal level, for some setting of $\lambda_1, \lambda_2 > 0$ describing the relative cost of storage and retrieval. What is the optimal division of labor between keeping information in working memory and recovering it through retrieval? The problem

$$\min_T \lambda_1 \sum_{t=1}^{T} tI_t + \lambda_2 \sum_{t=T+1}^{S} I_t \tag{32}$$

has solution $T \approx \frac{\lambda_2}{\lambda_1}$. This means that, as long as retrievals are more expensive than keeping the same amount of information in working memory (i.e., $\lambda_2 > \lambda_1$), the optimal strategy stores information from the last $T > 1$ words in working memory. Due to the factor t inside $\sum_{t=1}^{T} tI_t$, the bound (**??**) will be reduced when I_t decays faster, i.e., there is strong information locality.

The assumption that retrieving information is more difficult than storing it is reasonable for cue-based retrieval models, as retrieval suffers from similarity-based interference effects due to the unstructured nature of the storage (**?**). A model that maintains no information in its working memory, i.e. $H[m_t] = 0$, would correspond to a cue-based retrieval model that stores nothing in its buffers and control states, and relies entirely on retrieval to access past information. Given the nature of representations assumed in models (**?**), such a model would seem to be severely restricted in its ability to parse language.

1.5 Results for Language Production

Here we show results linking memory and locality in production. We show that results similar to our main theorem hold for the tradeoff between a speaker's memory and the accuracy with which they match the distribution of the language.

Speaker aims to match language distribution First, we consider a setting in which a speaker produces sentences with bounded memory, and analyze the deviation of the produced distribution from the actual distribution of the language.

We consider a speaker who maintains memory representations and incrementally produces based on these representations:

$$p_{speaker}(x_t|X_{<t}) = p(x_t|m_t) \tag{33}$$

We show a tradeoff between the memory capacity $\mathrm{H}[m_t]$ and the KL-divergence between the actual language statistics and the speaker's production distribution:

$$D_{KL}(P_{language}||P_{produced}) := \mathbb{E}_{X_{<t}} \sum_{x_t} p(x_t|X_{<t}) \log \frac{p(x_t|X_{<t})}{p_{speaker}(x_t|X_{<t})} \tag{34}$$

Theorem 3. *If a speaker maintains memory*

$$\mathrm{H}[m_t] \leq \sum_{i=1}^{T} tI_t \tag{35}$$

then

$$D_{KL}(P_{language}||P_{produced}) \geq \sum_{t=T+1}^{\infty} I_t \tag{36}$$

While this bound only considers the production of a single word, it immediately entails a bound on the production accuracy for sequences:

$$D_{KL}(P_{language}(X_1 \ldots X_t|X_{\leq 0})||P_{produced}(X_1 \ldots X_t|X_{\leq 0})) = t \cdot D_{KL}(P_{language}(X_1|X_{\leq 0})||P_{produced}(X_1|X_{\leq 0})) \tag{37}$$

Proof. First note

$$\begin{aligned}
D_{KL}(P_{language}||P_{produced}) &= \mathbb{E}_{X_{<t}} \sum_{x_t} p(x_t|X_{<t}) \log \frac{p(x_t|X_{<t})}{p_{speaker}(x_t|X_{<t})} && (38)\\
&= \mathbb{E}_{X_{<t}} \sum_{x_t} p(x_t|X_{<t}) \log \frac{p(x_t|X_{<t})}{p(x_t|M(X_{<t}))} && (39)\\
&= \mathbb{E}_{X_{<t}} \sum_{x_t} p(x_t|X_{<t}) \log p(x_t|X_{<t}) - \mathbb{E}_{X_{<t}} \sum_{x_t} p(x_t|X_{<t}) \log p(x_t|M(X_{<t})) && (40)\\
&= \mathbb{E}_{X_{<t}} \sum_{x_t} p(x_t|X_{<t}) \log p(x_t|X_{<t}) + S_M(x_t|X_{<t}) && (41)
\end{aligned}$$

In the last line, the first term is a constant independent of M.

Then the proof for the listener case transfers to this setting. □

TODO limitations of this as a model of production

Speaker aims to match, conditional on goal The first setting does not account for the fact that language is produced aiming for some communicative goal. We therefore now assume that the speaker has a communicative goal G in mind. This goal G stays constant during production process for a sentence, and we count how much memory is needed in addition to the goal G. We assume that there is a distribution of sentences expressing goals G:

$$P(sentence|G) \tag{42}$$

and assume that the speaker aims to match this distribution

$$\mathbb{E}_G[D_{KL}((language|G)||(produced|G))] \tag{43}$$

We can analyze this model by adding conditioning w.r.t. G throughout the analysis of the previous case. Specifically, we need

$$I_t^G := I[X_t, X_0|X_1, \ldots, X_{t-1}, G] \tag{44}$$

Take I_t conditioned on G: only count statistical dependencies to the degree that they are not redundant with the goal

Theorem 4. *If a speaker maintains memory*

$$\mathrm{H}[m_t] \leq \sum_{i=1}^{T} t I_t^G \tag{45}$$

then

$$\mathbb{E}_G D_{KL}(P_{language}(\cdot|G)||P_{produced}(\cdot|G)) \geq \sum_{t=T+1}^{\infty} I_t^G \tag{46}$$

Proof. This is entirely analogous to the previous proof. □

are there conditions under which this is close to I_t?

Pragmatic Speaker would need an assumption on the density of goals in the space of sequences.
Note

$$D_{KL}(P_{language}(x_{1\ldots t})||P_{produced}(x_{1\ldots t})) := \sum_{x_{1\ldots t}} p(x_t|X_{1\ldots t}) \log \frac{p(x_t|X_{1\ldots t})}{p_{speaker}(x_t|X_{1\ldots t})} \geq t D_K L(x_t||\ldots) \tag{47}$$

$H[G|Produced] - H[G|Language]$

2 Proof of Left-Right Invariance

Here we show that the bound provided by our theorem is invariant under reversal of the process. That is: Given a process $(X_t)_{t\in\mathbb{Z}}$, we define its reverse process $(Y_t)_{t\in\mathbb{Z}}$ by $Y_t := X_t$. We claim that the theorem provides the same bounds for the memory-surprisal tradeoff curves. To prove this, we note:

$$I[X_t, X_0|X_{1\ldots t-1}] = I[Y_{-t}, Y_0|Y_{1-t\cdots-1}] = I[Y_0, Y_t|Y_{1\ldots t-1}] = I[Y_t, Y_0|Y_{1\ldots t-1}] \tag{48}$$

The first step follows from the definition of Y. The second step follows from the fact that X_t, and thus also Y_t, is stationary, and thus adding t to each index in the expression does not change the resulting value. The third step uses the fact that mutual information is symmetric.

3 Examples with Analytical Calculations

Here, we provide examples of the theorem in settings where analytical calculations are possible.

3.1 Example I: Tight Bounds in Artificial Language

Here we provide explicit calculations for the artificial language simulation, showing that the bounds provided by the theorem are tight in this case.

TODO

3.2 Example II: Window-Based Model not Optimal

Here we provide an example of a stochastic process where a window-based memory encoding is not optimal, but the bound provided by our theorem still holds. This is an example where the bound provided by the theorem is loose: while it bounds the memory-surprisal tradeoff of all possible listeners, the bound is 'optimistic', meaning that no mathematically possible memory encoding function M can exactly achieve the bound.

Let k be some positive integer. Consider a process $x_{t+1} = (v_{t+1}, w_{t+1}, y_{t+1}, z_{t+1})$ where

1. The first two components consist of fresh random bits. Formally, v_{t+1} is an independent draw from $Bernoulli(0.5)$, independent from all preceding observations $x_{\leq t}$. Second, let w_{t+1} consist of $2k$ many such independent random bits (so that $H[w_{t+1}] = 2k$)

2. The third component *deterministically* copies the first bit from $2k$ steps earlier. Formally, y_{t+1} is equal to the first component of x_{t-2k+1}

3. The fourth component *stochastically* copies the second part (consisting of $2k$ random bits) from one step earlier. Formally, each component $z^{(i)}_{t+1}$ is determined as follows: First take a sample $u^{(i)}_{t+1}$ from $Bernoulli(\frac{1}{4k})$, independent from all preceding observations. If $u^{(i)}_{z+1} = 1$, set $z^{(i)}_{t+1}$ to be equal to the second component of $w^{(i)}_t$. Otherwise, let $z^{(i)}_{t+1}$ be a fresh draw from $Bernoulli(0.5)$.

Predicting observations optimally requires taking into account observations from the $2k$ last time steps.

We show that, when approximately predicting with low memory capacities, a window-based approach does *not* in general achieve an optimal memory-surprisal tradeoff.

Consider a model that predicts x_{t+1} from only the last observation x_t, i.e., uses a window of length one. The only relevant piece of information in this past observation is w_t, which stochastically influences z_{t+1}. Storing this costs $2k$ bit of memory as w_t consists of $2k$ draws from $Bernoulli(0.5)$. How much does it reduce the surprisal of x_{t+1}? Due to the stochastic nature of z_{t+1}, it reduces the surprisal only by about $I[x_{t+1}, w_t] = I[z_{t+1}, w_t] < 2k \cdot \frac{1}{2k} = 1$, i.e., surprisal reduction is strictly less than one bit. [3]

We show that there is an alternative model that strictly improves on this window-based model: Consider a memory encoding model that encodes each of $v_{t-2k+1}, \ldots, v_t$, which costs $2k$ bits of memory – as the window-based model did. Since $y_{t+1} = v_{t-2k+1}$, this model achieves a surprisal reduction of $H[v_{t-2k+1}] = 1$ bit, strictly more than the window-based model.

This result does not contradict our theorem because the theorem only provides *bounds* across models, which are not necessarily achieved by a given window-based model. In fact, for the process described here, no memory encoding function M can exactly achieve the theoretical bound described by the theorem.

[3] We can evaluate $I[z_{t+1}, w_t]$ as follows. Set $l = k/4$. Write z, w for any of the $2k$ components of z_{t+1}, w_t, respectively. First, calculate $p(z = 1|w = 1) = 1/l + (1 - 1/l)\frac{1}{2} = 1/(2l) + 1/2 = \frac{1+l}{2l}$ and $p(z = 0|w = 1) = (1 - 1/l)\frac{1}{2} = 1/2 - 1/2l = \frac{l-1}{2l}$. Then $I[Z, W] = D_{KL}(p(z|w = 1)||p(z)) = \frac{1+l}{2l} \log \frac{\frac{1+l}{2l}}{1/2} + \frac{l-1}{2l} \log \frac{\frac{l-1}{2l}}{1/2} = \frac{1+l}{2l} \log \frac{1+l}{l} + \frac{l-1}{2l} \log \frac{l-1}{l} \leq \frac{1+l}{l} \log \frac{1+l}{l} = (1 + 1/l) \log(1 + 1/l) \leq (1 + 1/l)(1/l) = 1/l + 1/l^2 < 2/l = \frac{1}{2k}$.

3.3 Example III: Tight Bound for Retrieval Model

Here, we provide an example where our bound is tight for the retrieval-based model even though it is quite loose for the capacity model. That means, while no memory encoding function can exactly achieve the bound in the capacity-bounded setting, there are memory encoding functions that exactly achieve the bound in the retrieval-based setting.

Let k be a positive integer. Consider a process $x_{t+1} = (y_{t+1}, z_{t+1}, u_{t+1}, v_{t+1})$ where

1. y_{t+1} consists of $2k$ random bits.
2. z_{t+1} is a draw from $Bernoulli(\frac{1}{4k})$.
3. u_{t+1} consists of $2k$ random bits if $z_t = 0$ and is equal to y_{t-2k+1} else.
4. $v_{t+1} := z_t$

Predicting observations x_{t+1} optimally requires storing $y_{t-2k+1}, \ldots, y_t$ and z_t. This amounts to $(2k+1) \cdot 2k + H_2[1/4k]$ bits of memory in the capacity-based model. However, $I[u_{t+1}, y_{t-2k+1} | z_{t+1}] \leq 1/k$ (TODO). Therefore, the theorem bounds the memory cost only by $HM \geq \sum_{t=1}^{\infty} tI_t = 1$. The bound provided by the theorem is therefore loose in this case.

However, it is tight for the retrieval-based model: We use s_t to store $y_{t-2k+1}, \ldots, y_t$, and we use the working memory m_{t+1} to store z_t. Then, if $z_t = 1$, we retrieve $r_t = g(x_{t-1}, m_{t-1}, s_{t-1}) := y_{t-2k+1}$. The cost of storing z_t is $H_2[1/4k]$, and the cost of retrieving r_t is $\frac{1}{4k} \cdot 2k$.

In total, $H[m_t] = H_2[1/4k]$ and $H[r_t] = 1/k$.

Taking, in the theorem, $T = 1$ and $S \to \infty$, we obtain $H[m_t] \geq H_2[1/4k]$ and $H[r_t] \geq 1/k$. Thus, the bound is tight for both working memory and retrieval costs.

4 Corpus Size per Language

Language	Training	Held-Out	Language	Training	Held-Out
Afrikaans	1,315	194	Indonesian	4,477	559
Amharic	974	100	Italian	17,427	1,070
Arabic	21,864	2,895	Japanese	7,164	511
Armenian	514	50	Kazakh	947	100
Bambara	926	100	Korean	27,410	3,016
Basque	5,396	1,798	Kurmanji	634	100
Breton	788	100	Latvian	4,124	989
Bulgarian	8,907	1,115	Maltese	1,123	433
Buryat	808	100	Naija	848	100
Cantonese	550	100	North Sami	2,257	865
Catalan	13,123	1,709	Norwegian	29,870	4,639
Chinese	3,997	500	Persian	4,798	599
Croatian	7,689	600	Polish	6,100	1,027
Czech	102,993	11,311	Portuguese	17,995	1,770
Danish	4,383	564	Romanian	8,664	752
Dutch	18,310	1,518	Russian	52,664	7,163
English	17,062	3,070	Serbian	2,935	465

Erzya	1,450	100	Slovak	8,483	1,060
Estonian	6,959	855	Slovenian	7,532	1,817
Faroese	1,108	100	Spanish	28,492	3,054
Finnish	27,198	3,239	Swedish	7,041	1,416
French	32,347	3,232	Thai	900	100
German	13,814	799	Turkish	3,685	975
Greek	1,662	403	Ukrainian	4,506	577
Hebrew	5,241	484	Urdu	4,043	552
Hindi	13,304	1,659	Uyghur	1,656	900
Hungarian	910	441	Vietnamese	1,400	800

Table 2: Languages, with the number of training and held-out sentences available.

5 Samples Drawn per Language

Language	Base.	Real	Language	Base.	Real
Afrikaans	13	10	Indonesian	11	11
Amharic	137	10	Italian	10	10
Arabic	11	10	Japanese	25	15
Armenian	140	76	Kazakh	11	10
Bambara	25	29	Korean	11	10
Basque	15	10	Kurmanji	338	61
Breton	35	14	Latvian	308	178
Bulgarian	14	10	Maltese	30	24
Buryat	26	18	Naija	214	10
Cantonese	306	32	North Sami	335	194
Catalan	11	10	Norwegian	12	10
Chinese	21	10	Persian	25	12
Croatian	30	17	Polish	309	35
Czech	18	10	Portuguese	15	55
Danish	33	17	Romanian	10	10
Dutch	27	10	Russian	20	10
English	13	11	Serbian	26	11
Erzya	846	167	Slovak	303	27
Estonian	347	101	Slovenian	297	80
Faroese	27	13	Spanish	14	10
Finnish	83	16	Swedish	31	14
French	14	11	Thai	45	19
German	19	13	Turkish	13	10
Greek	16	10	Ukrainian	28	18
Hebrew	11	10	Urdu	17	10
Hindi	11	10	Uyghur	326	175
Hungarian	220	109	Vietnamese	303	12

Figure 1: Samples drawn per language according to the precision-dependent stopping criterion.

Language	Mean	Lower	Upper	Language	Mean	Lower	Upper
Afrikaans	1.0	1.0	1.0	Indonesian	1.0	1.0	1.0
Amharic	1.0	1.0	1.0	Italian	1.0	1.0	1.0
Arabic	1.0	1.0	1.0	Japanese	1.0	1.0	1.0
Armenian	0.92	0.87	0.97	Kazakh	1.0	1.0	1.0
Bambara	1.0	1.0	1.0	Korean	1.0	1.0	1.0
Basque	1.0	1.0	1.0	Kurmanji	0.93	0.88	0.98
Breton	1.0	1.0	1.0	Latvian	0.49	0.4	0.57
Bulgarian	1.0	1.0	1.0	Maltese	1.0	1.0	1.0
Buryat	1.0	1.0	1.0	Naija	1.0	0.99	1.0
Cantonese	0.96	0.86	1.0	North Sami	0.37	0.3	0.44
Catalan	1.0	1.0	1.0	Norwegian	1.0	1.0	1.0
Chinese	1.0	1.0	1.0	Persian	1.0	1.0	1.0
Croatian	1.0	1.0	1.0	Polish	0.1	0.04	0.17
Czech	1.0	1.0	1.0	Portuguese	1.0	1.0	1.0
Danish	1.0	1.0	1.0	Romanian	1.0	1.0	1.0
Dutch	1.0	1.0	1.0	Russian	1.0	1.0	1.0
English	1.0	1.0	1.0	Serbian	1.0	1.0	1.0
Erzya	0.99	0.98	1.0	Slovak	0.07	0.03	0.12
Estonian	0.8	0.72	0.86	Slovenian	0.82	0.77	0.88
Faroese	1.0	1.0	1.0	Spanish	1.0	1.0	1.0
Finnish	1.0	1.0	1.0	Swedish	1.0	1.0	1.0
French	1.0	1.0	1.0	Thai	1.0	1.0	1.0
German	1.0	0.91	1.0	Turkish	1.0	1.0	1.0
Greek	1.0	1.0	1.0	Ukrainian	1.0	1.0	1.0
Hebrew	1.0	1.0	1.0	Urdu	1.0	1.0	1.0
Hindi	1.0	1.0	1.0	Uyghur	0.65	0.57	0.73
Hungarian	0.87	0.8	0.93	Vietnamese	1.0	0.98	1.0

Figure 2: Bootstrapped estimates for G.

6 Detailed Results per Language

7 Details for Neural Network Models

8 N-Gram Models

8.1 Method

We use a version of Kneser-Ney Smoothing. For a sequence $w_1 \ldots w_k$, let $N(w_{1\ldots k})$ be the number of times $w_{1\ldots k}$ occurs in the training set. The unigram probabilities are estimated as

$$p_1(w_t) := \frac{N(w_t) + \delta}{|Train| + |V| \cdot \delta} \tag{49}$$

where $\delta \in \mathbb{R}_+$ is a hyperparameter. Here $|Train|$ is the number of tokens in the training set, $|V|$ is the number of types occurring in train or held-out data. Higher-order probabilities $p_t(w_t|w_{0\ldots t-1})$ are estimated recursively as follows. Let $\gamma > 0$ be a hyperparameter. If $N(w_{0\ldots t-1}) < \gamma$, set

$$p_t(w_t|w_{0\ldots t-1}) := p_{t-1}(w_t|w_{1\ldots t-1}) \tag{50}$$

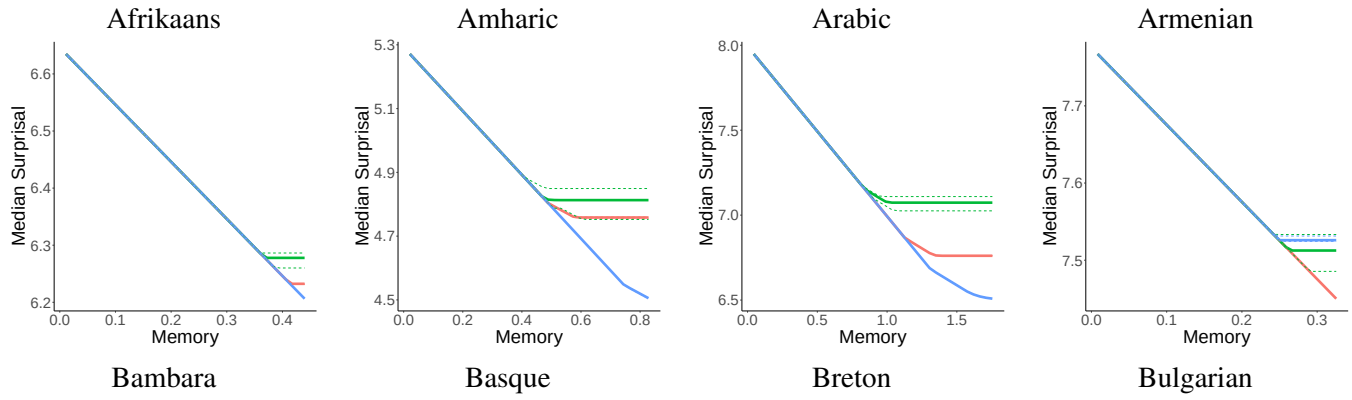
Otherwise, we interpolate between t-th order and lower-order estimates:

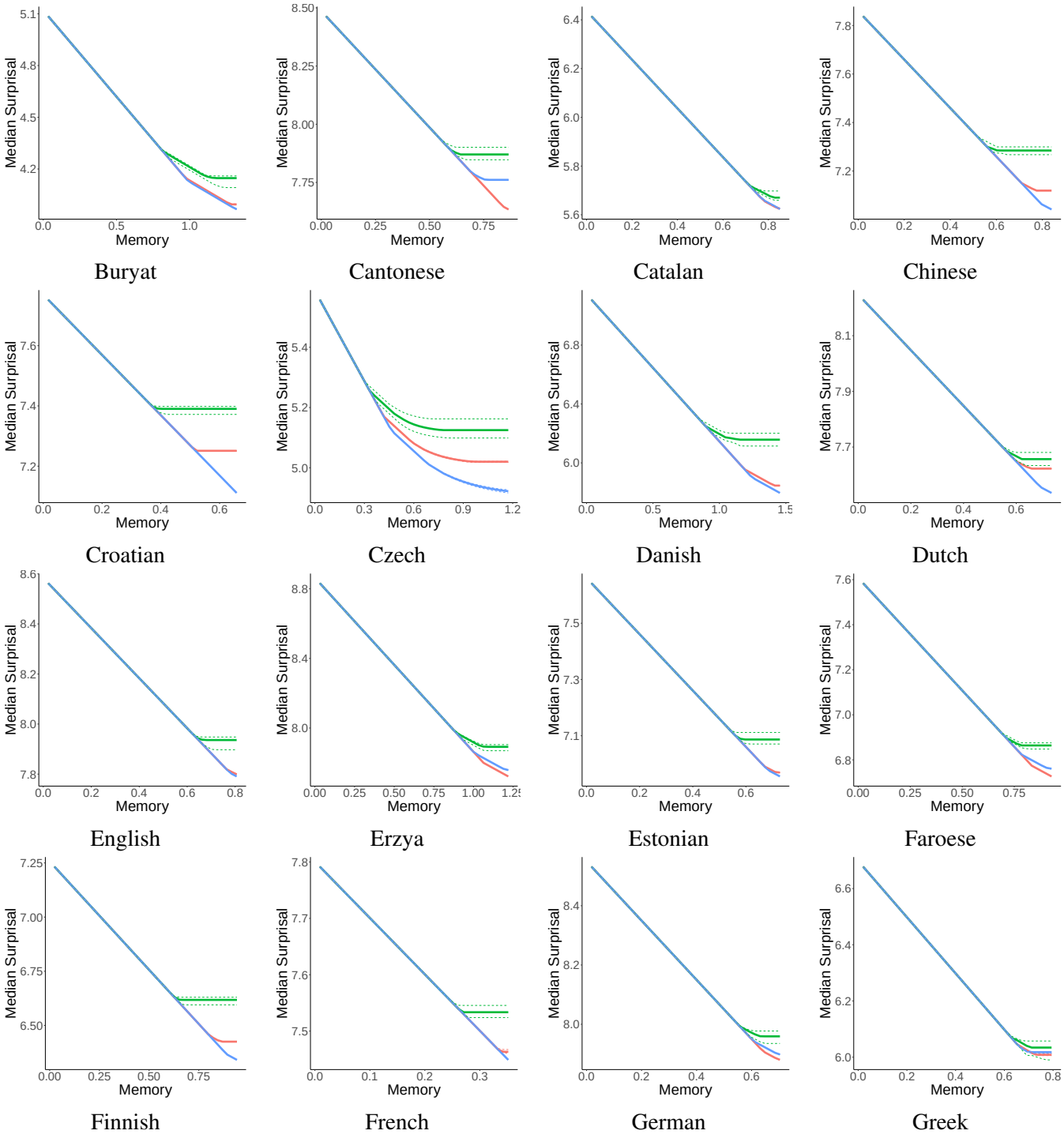
$$p_t(w_t|w_{0\ldots t-1}) := \frac{\max(N(w_{0\ldots t}) - \alpha, 0.0) + \alpha \cdot \#\{w : N(w_{0\ldots t-1}w) > 0\} \cdot p_{t-1}(w_t|w_{1\ldots t-1})}{N(w_{0\ldots t-1})} \tag{51}$$

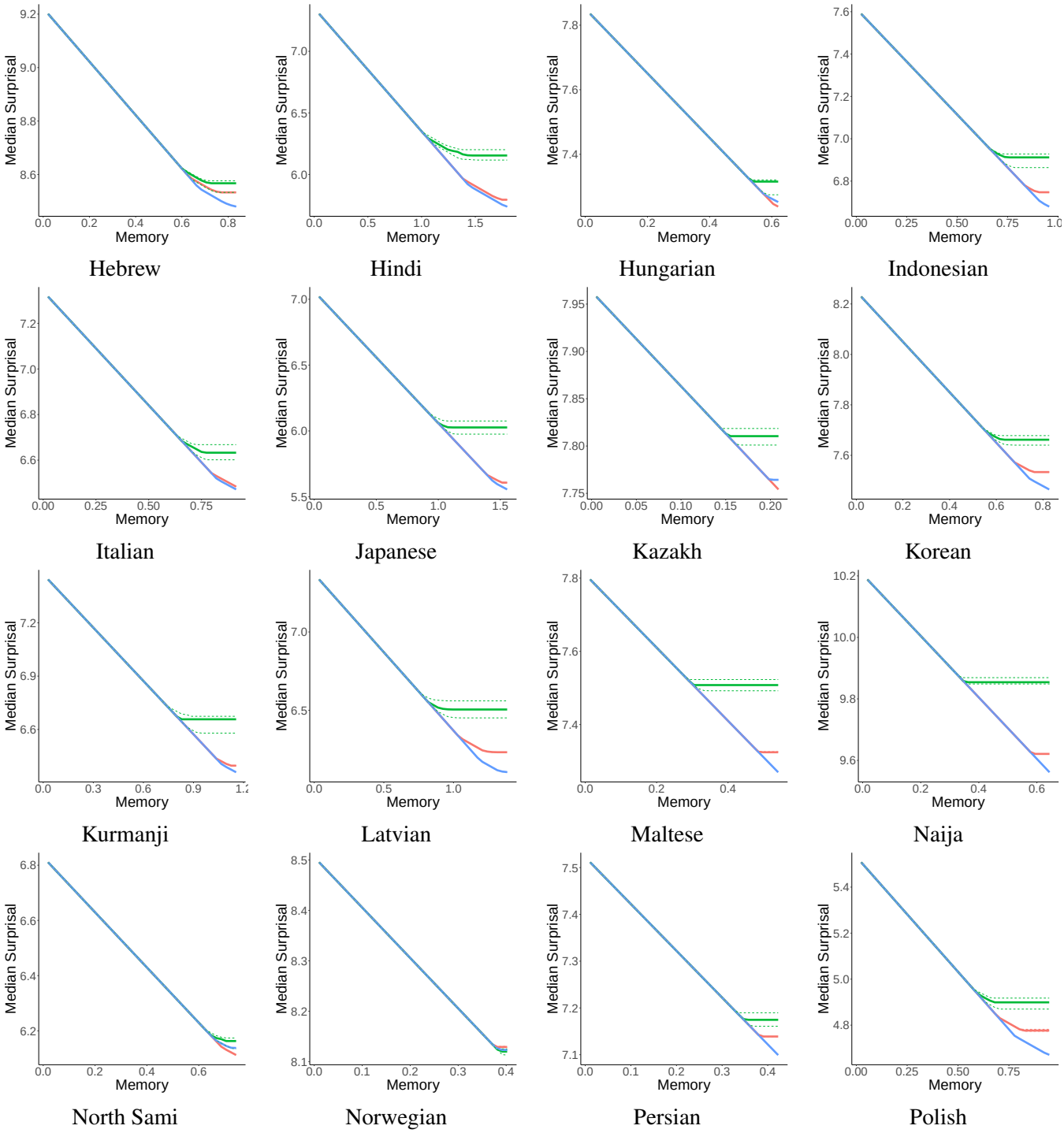
where $\alpha \in [0,1]$ is also a hyperparameter. (CITE) show that this definition results in a well-defined probability distribution, i.e., $\sum_{w \in V} p_t(w|w_{0\ldots t-1}) = 1$.

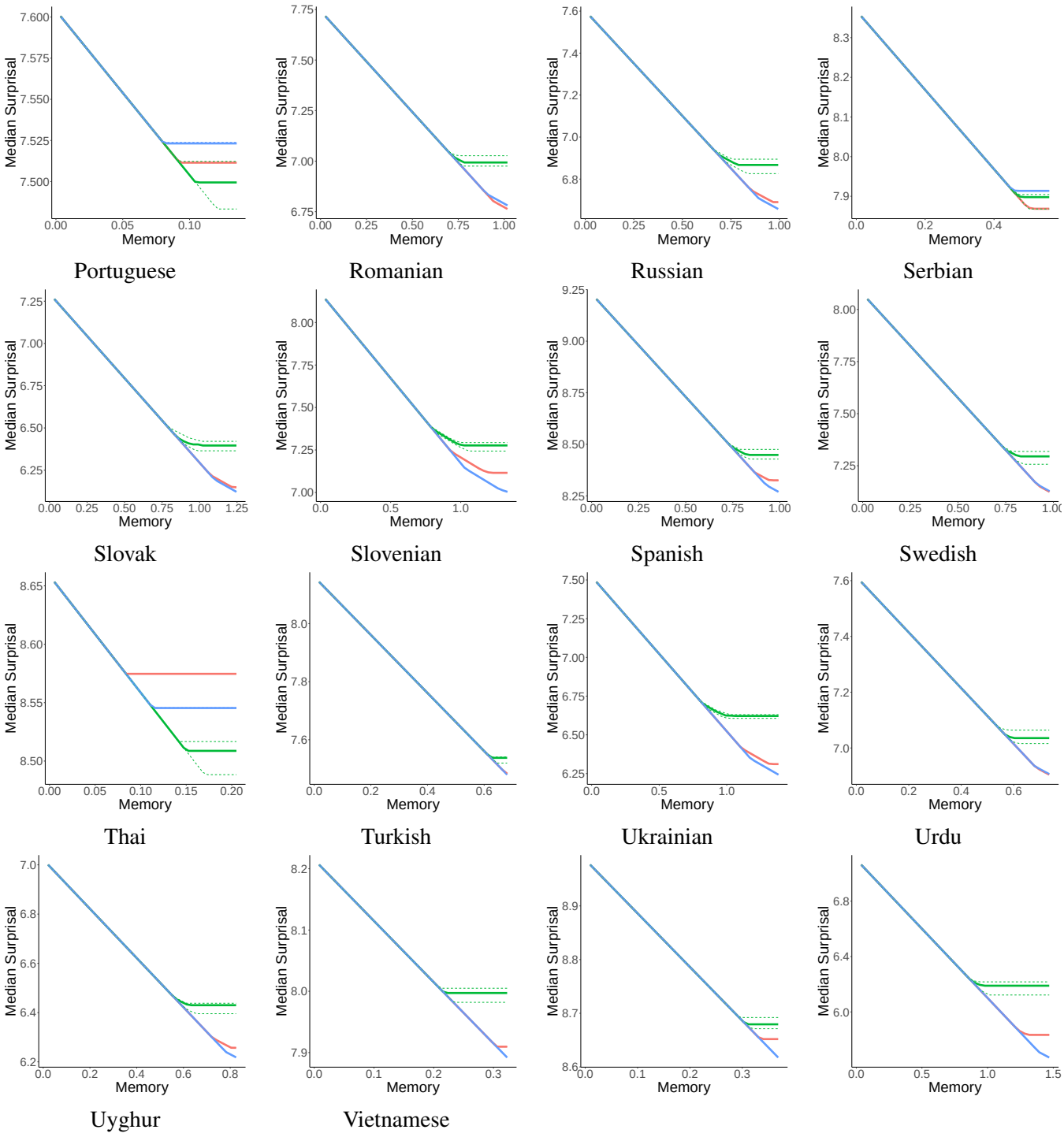
Hyperparameters α, γ, δ are tuned with the same strategy as for the neural network models.

8.2 Results




Median Surprisal
Memory
Buryat
Cantonese
Catalan
Chinese
Croatian
Czech
Danish
Dutch
English
Erzya
Estonian
Faroese
Finnish
French
German
Greek


Median Surprisal
Memory
Hebrew
Hindi
Hungarian
Indonesian
Italian
Japanese
Kazakh
Korean
Kurmanji
Latvian
Maltese
Naija
North Sami
Norwegian
Persian
Polish


Median Surprisal
Memory
Portuguese
Romanian
Russian
Serbian
Slovak
Slovenian
Spanish
Swedish
Thai
Turkish
Ukrainian
Urdu
Uyghur
Vietnamese

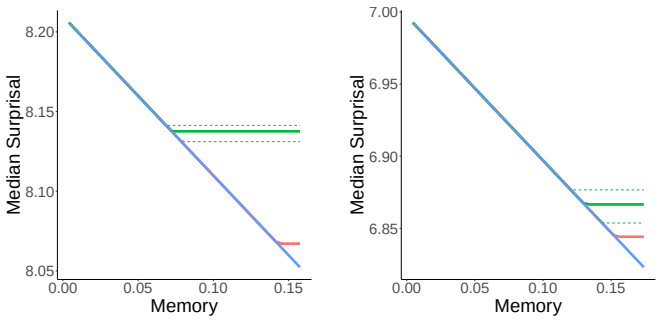


Figure 3: Medians (estimated using n-gram models): For each memory budget, we provide the median surprisal for real and random languages. Solid lines indicate sample medians for ngrams, dashed lines indicate 95 % confidence intervals for the population median. Green: Random baselines; blue: real language; red: maximum-likelihood grammars fit to real orderings.

9 Chart Parsing Control

LSTMs and n-gram models are linear sequence models that might incorporate biases towards linear order as opposed to hierarchical structure. Here we use chart parsing to show that the results also hold when estimating I_t using a model that is based on hierarchical structure and incorporates no bias towards linear closeness.

We use PCFGs. PCFG surprisal is often computed in psycholinguistic research using approximate incremental parsers, but these might themselves incorporate some biases towards linear closeness. We instead opt for exact inference for PCFGs using exact chart parsing.

9.1 Deriving PCFGs from Dependency Corpora

We binarize, and give assign nonterminal labels to intermediate nodes based on (1) the POS of the head, (2) the lexical identity of the head, (3) the dependency label linking head and dependent. We binarize so that left children branch off before right children.

The preterminals are labeled by POS tag and lexical identity.

It is necessary to reduce the number of preterminals and nonterminals, both to deal with data sparsity, and to make chart parsing tractable. In our implementation for calculating I_t (see below), we found that up to 700 nonterminals were compatible with efficient inference. (For comparison, the Berkeley parser uses X nonterminals for its English grammar, but employs a highly optimized coarse-to-fine strategy.)

We reduced the number of nonterminals as follows: (1) For words with frequency below a threshold parameter, we did not record lexical identity in preterminals and nonterminals. (2) Nonterminals that only differ in the relation label were merged if their frequency fell below a threshold parameter, (2) Nonterminals that only differ in the head's lexical identity were merged if their frequency fell below a threshold parameter.

Words occurring less than 3 times in the dataset were replaced by OOV.

Alternative: merge-and-split, but that would have taken too long to run on all the corpora.

We chose the threshold parameters for (1)-(3) separately for each language by sampling 15 configurations, and choosing the one that minimized estimated surprisal (see below) on a sampled baseline grammar, while resulting in at most 700 nonterminals and preterminals.

mention approaches in the literature

An alternative avoiding binarization would be to use the Earley parser, but that would have made it less feasible to parallelize processing on GPUs (see below).

9.2 Estimating I_t with Chart Parsing

algorithm, cite Goodman's thesis

Calculating I_t requires estimating entropies $H[X_1,\dots,X_t]$, and thus probabilities $P(X_1,\dots,X_t)$. This is challenging because it requires marginalization over possible positions in a sequence.

There is a known extension of the CKY algorithm that calculates *prefix* probabilities

$$P[\#,X_1,\dots,X_t] := \sum_N \sum_{Y_{1\dots N}} P(\#,X_1,\dots,X_t,Y_{1\dots N},\#) \tag{52}$$

(here, # denotes the beginning/end of a sentence), that is, the probability mass assigned to all sentences starting with the given prefix $X_1,\dots,X_t$.

However, simultaneously summing over possible left *and* right continuations is more challenging.[4] We approach this by restricting the summation on the left to prefixes of a fixed length:

$$\sum_{Y_1\dots Y_N} P(\#,Y_1\dots Y_N,X_1,\dots,X_t) \tag{53}$$

and estimating

$$P(X_t|X_1\dots X_{t-1}) \approx \mathbb{E}_{Y_1\dots Y_N} P(X_t|\#,Y_1\dots Y_N,X_1,\dots,X_{t-1}) \tag{54}$$

Under certain conditions on the PCFG, this approximation converges to the tru value for sufficiently large values of N.[5] Empirically, we found that the values already became essentially stationary at $N \geq 5$.

The resulting algorithm is shown in X.

For computational efficiency, we estimated I_t for $t = 1,\dots 5$, finding I_t to be very close to zero for higher t.

We ran the algorithm on all contiguous sequences of length $T = 5$. Following (CITE), we took advantage of GPU parallelization for implementation, processing 1,000 sequences in parallel.

9.3 Results

We computed I_t for the MLE grammar and for five random baseline grammars.

We did not run this on the observed orderings, as these may have crossing branches, making binarization difficult and thus rendering comparison with baselines less straightforward.

Figure **??**

Limitations: Absolute numbers aren't comparable with other models because there are many OOVs (they are necessary because the number of non- and preterminals has to be kept low). Also, the amount of exploited predictive information $\sum_t I_t$ is much lower than in the other models. Agrees with the observation that PCFG independence assumptions are inadequate, and that chart parsers have not historically reached good perplexities (parsers with good perplexities such as Roark Parser and RNNGs do not make these

[4] (CITE) describe a method for calculating infix probabilities, but that method computes something subtly different from the quantity required here, and it is also computationally costly.

[5] TODO say something about Markov chain convergence: For each t, consider for each nonterminal τ the number n_τ of nodes with this nonterminal dominating w_τ. This is a Markov chain.

independence assumptions, but also do not allow efficient exact chart parsing). Nonetheless, the experiment confirms the finding with a model that is based on hierarchical syntactic structure while enabling exact inference.

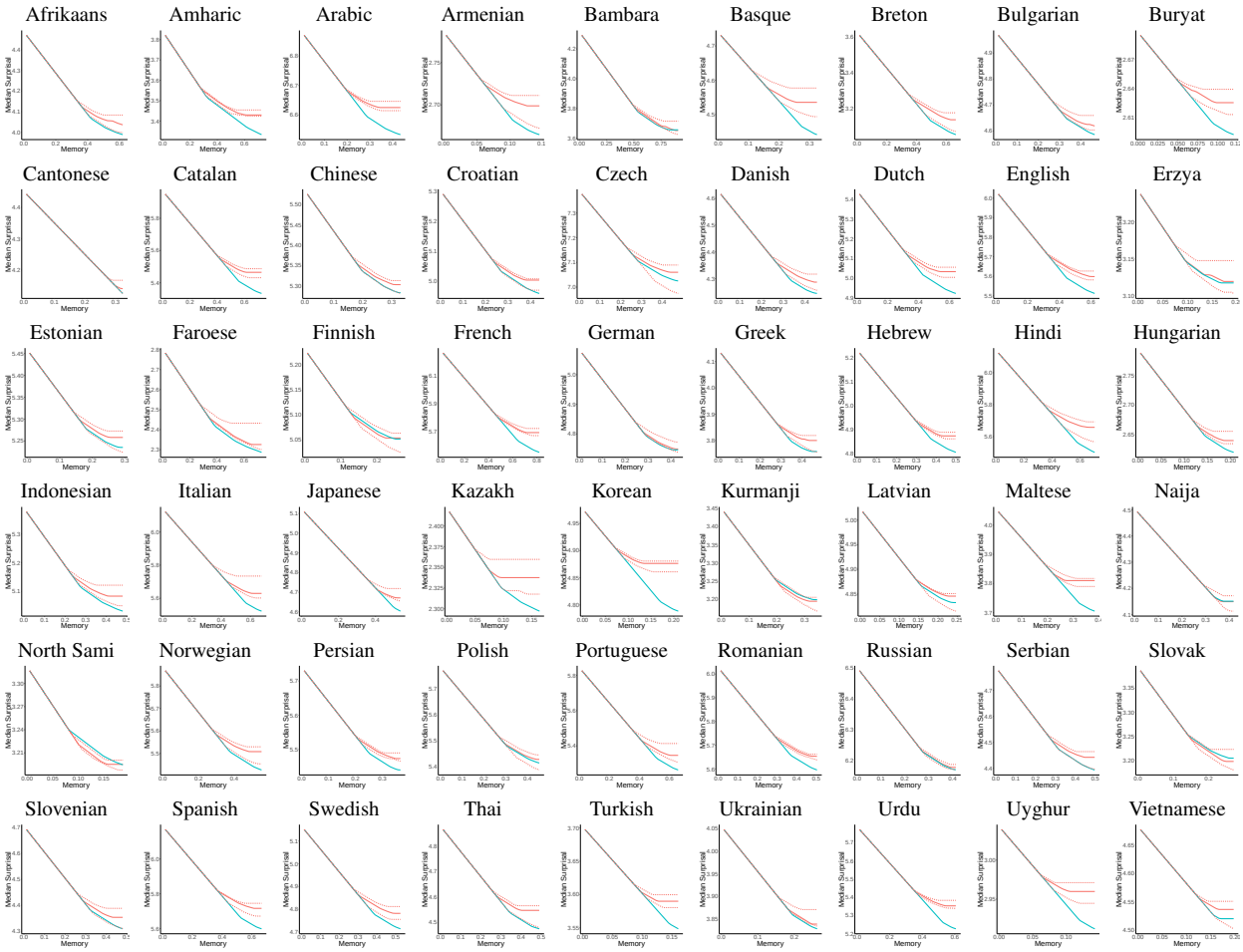


Figure 4: PCFG estimator, comparing fitted grammars (blue) with baselines (red)

10 Morphology

10.1 Japanese

Here, we describe how we determined the Japanese verb suffixes described in the main paper.

We determined a set of frequent morphemes as follows. We selected all morphemes occurring in the dataset at least 50 times and annotated their meaning/function. Among these, three morphemes are treated as independent words, not suffixes, by **?** (*dekiru* 'be able to', *naru* 'become', *yoo* 'as if'); we excluded these. Furthermore, passive and potential markers are formally identical for many verbs; we included both here.

We list the morphemes according to the order extracted according to the model.

Note that there is no universally accepted segmentation for Japanese suffixes; we follow the UD tok-

enization in choosing which suffixes to segment.[6]

1. VALENCE: causative (*-(s)ase-*). (**?**, 142) (**?**, Chapter 13). In the UD data, this is lemmatized as *saseru*, *seru* (190 occurrences).
2. VOICE: passive (*-are-*, *-rare-*) (**?**, 152) (**?**, Chapter 12) In the UD data, this is lemmatized as *rareru*, *reru* ($\approx$ 2000 occurrences).
3. MOOD, MODALITY:
   (a) potential (allomorphs *-are-*, *-rare-*, *-e-*). In the UD data, this is lemmatized as *rareru*, *reru*, *eru*, *keru*. This is formally identical to the passive morpheme for many verbs ((**?**, 346), (**?**, 398)).
   (b) politeness *-mas-* (allomorphs *-masu-*, *-mashi-*, *-mase-*) (**?**, 190). In the UD data, this is lemmatized as *masu* ($\approx$ 600 occurrences).
   (c) MODALITY: desiderative *-ta-* (allomorphs: *-tai*, *-taku-*, *-taka-*) (85 occurrences) (**?**, 238).
4. NEGATION: negation *-na-* (allomorphs: *-nai*, *-n-*, *-nakat-*). Lemmatized as *nai* (630 occurrences).
5. TENSE/ASPECT/MOOD:
   (a) *-ta* for past (4K occurrences) (**?**, 211)
   (b) *-yoo* for hortative, future, and similar meanings (**?**, 229). This is lemmatized as *u* (92 occurrences).
6. *-te* derives a nonfinite form (**?**, 186). (4K occurrences)

We provide examples illustrating the relative ordering of different morphemes. Note that passive and potential markers do not co-occur. We omit examples with -te; it always follows other suffixes that are compatible with it.

Stem	Caus.	Pass.	Pot.	Polite.	Desid.	Neg.	TAM	
mi						naka	tta	did not see (**?**, 153)
mi					taku	nai		I do not wish to see (**?**, 98)
mi					taku	naka	tta	I did not wish to see (**?**, 98)
tat	ase	rare					ta	was made to stand up (**?**, 396)
waraw		are					ta	was laughed at (**?**, 384)
mi		rare		mase		n		is not seen (**?**, 337)
mi		rare		mash			yoo	will be seen (**?**, 337)
de						naka	roo	will not go out (**?**, 170)
mi			e	mase		n		cannot see (**?**, 349)

10.2 Sesotho

Here, we describe how we determined the Sesotho verb prefixes and suffixes.

Sesotho has composite forms consisting of an inflected auxiliary followed by an inflected verb. Both verbs carry subject agreement. While they are annotated as a unit in the Demuth corpus, they are treated as

[6]The biggest difference to some other treatments is that the ending *-u*/*-ru* is viewed as part of the preceding morpheme that appears in some environments due to allomorphic variation, while it is viewed as a nonpast suffix in some other treatments (**?**, p.116).

separate words in grammars (**??**). We separated these, taking the main verb to start at its subject agreement prefix. We only considered main verbs for the experiments here.

Forms in child utterances are annotated with well-formed adult forms; we took these here.

In the Demuth corpus, each morpheme is annotated; a one- or two-letter key indicates the type of morpheme (e.g. subject agreement, TAM marker). We classified morphemes by this annotation.

According to **?**, affixes in the Sesotho verb have the following order:

1. Subject agreement
2. Tense/aspect
3. Object agreement
4. Verb stem
5. 'Extension'/perfect/passive markers, where 'extension' refers to causative, neuter/stative, reversive, etc.
6. Mood

We refined this description by considering all morpheme types occurring at least 50 times in the corpus.

As in Japanese, morphemes show different forms depending on their environment, and the corpus contains some instances of fused neighboring morphemes that were not segmented further.

Prefixes

1. Subject agreement:

   This morpheme encodes agreement with the subject, for person, number, and noun class (the latter only in the 3rd person) (**?**, §395) (**?**, p. 162).

   In the Demuth corpus, this is annotated as *sm* (17K occurrences) for ordinary forms, and *sr* (193 occurrences) for forms used in relative clauses.

2. Negation:

   In various TAM forms, negation is encoded with a morpheme *-sa-* in this position (362 occurrences) (**?**, p. 172) (**?**, §429). Common allomorphs in the corpus include *ska*, *seka*, *sa*, *skaba*.

3. Tense/Aspect/Mood (13K occurrences)

   Tense/aspect marker (tˆ 13K) (**?**, p. 165)

   Common TAM markers in this position in the corpus include, with the labels provided in the Demuth corpus:

   - -tla-, -tlo-, -ilo- future (**?**, §410–412)
   - -a- present (**?**, §400)
   - -ka- potential (**?**, §422–428)
   - -sa- persistive (**?**, §413–418)
   - -tswa- recent past (**?**, §404–406)

In the corpus, TAM prefixes are often fused with the subsequent object marker.

4. OBJECT agreement (labeled om, 6K occurrences) or reflexive (labeled rf, 751 occurrences).

   Similar to subject agreement, object agreement denotes person, number, and noun class features of the object. Unlike subject agreement, it is optional (**?**, §459).

   Object agreement and reflexive marking are mutually exclusive (**?**, p. 165).

In addition to these morphemes used in finite verbs, there is an *infinitive* prefix *ho-* (labeled 'if', 314 occurrences) (**?**, §§378-384), which is compatible with TAM markers (**?**, §379) and object agreement (**?**, §382).

Verb Suffixes in Sesotho Again, we extracted morpheme types occurring at least 50 times.

1. Reversive: (labeled *rv*, 214 occurrences), (**?**, §345).

   This suffix changes semantics. Examples: tlama 'bind' – tlamōlla 'loosen', etsa 'do' – etsōlla 'undo' (**?**, §346). Such suffixes are found across Bantu languages **?**.

2. VALENCE:

   (a) causative (labeled *c*, 1K occurrences), *-isa* (with morphophonological changes) (**?**, §325)

   (b) neuter (labeled *nt*, 229 occurrences), *-eha*, *-ahala* (**?**, §307)
   The neuter suffix reduces valence: lahla 'throw away' – lahlela 'get lost', sēnya 'to damage' – sēnyeha 'to get damaged' (**?**, §308).

   (c) applicative (labeled *ap*, 2K occurrences) -el- (**?**, §310)
   The applicative suffix increases valence: bōlela 'to say' bōlella 'to say to (s.o.)' (**?**, §310).

   (d) Perfective/Completive *-ella* (annotated cl, 66 occurrences) (**?**, §336)
   This does not actually change valence, but it is formally a reduplication of the applicative suffix (**?**, §336), and as such its ordering behavior patterns with that of valence suffixes, in particular, it is placed before the passive suffix.[7]

   (e) Reciprocal *-ana* (annotated rc, 103 times) (**?**, §338)
   This reduces valence: rata 'to love' – ratana 'to love another' (**?**, §338).

   Some of these suffixes can be stacked, e.g., see (**?**, §345) for reversive+causative, and (**?**, §314-315) for applicative suffixes applied to other valence affixes.[8]

   Some other suffixes documented in the literature do not occur frequently or are not annotated in the corpus (e.g., the associative suffix (**?**, textsection 343)).

3. VOICE: passive *-w-* (labeled *p*, 1K occurrences) (**?**, §300)

4. TENSE: tense (labeled tˆ, 3K occurrences) .

   The only tense suffix is the perfect affix *-il-*, which has a range of allomorphs depending on the preceding stem and valence/voice suffixes, if present (**?**, §369), (**?**, p. 167). Common morphs in the Demuth corpus are -il- and -its-.

[7] Example from the Demuth corpus: u-neh-el-ets-w-a-ng tˆp.om2s-give-ap-cl-p-mˆin-wh 'What is it that you want passed to you?'.

[8] Example of reciprocal+applicative from Demuth corpus: ba-arol-el-an-a sm2-tˆp_divide-ap-rc-mˆin 'Do they share?'

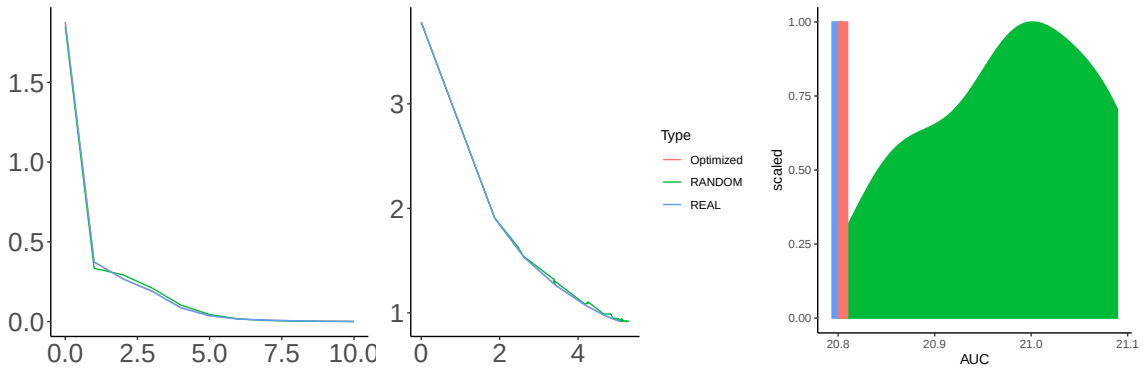


Figure 5: Japanese verb suffixes, measuring prediction on the level of phonemes, for real (blue), random (green), and approximately optimized (red) orderings. Left: I_t as a function of t. Center: Memory-surprisal tradeoff. Right: Areas under the curve for the memory-surprisal tradeoff.

5. MOOD: Mood (labeled mˆ, 37K occurrences)

   In the Demuth corpus, the following mood endings are labeled (the analysis provided by **?** is different from that provided by **?**, meaning the citations are only approximate):

   (a) Imperative (labeled IMP) (**?**, §386–387): singular (-e, labeled IMP) (**?**, §386) and plural (-ang, labeled IMP.PL) (**?**, §386).

   Similar subjunctive SBJV1 *-e* (singular), *-eng* (plural).

   (b) IND (-a, -e) and NEG (-e, -a) (**?**, §394–421).

   (c) subjunctive SBJV2 (-e, -a) (**?**, §444–455)

6. Interrogative (labeled *wh*, 2K times) and relative (labeled *rl*, 857 times) markers *-ng*.

   The interrogative marker *-ng* is a clitic form of *eng* 'what' according to (**?**, p. 168), (**?**, §160, 320, 714); it is treated as a suffix in the Demuth corpus.

   The relative marker *-ng* is affixed to verbs in relative clauses are marked with -ng (**?**, §271, 793).

Examples from **?**:

Sbj.	Ng.	TAM	Obj.	V	Val.	Voice	Tense	Mood	
o				pheh			il	e	(Thabo) cooked (food) (**?** (15))
ke			e	f		uw		e	(I) was given (the book) (**?** (26c))
o				pheh	el			a	(Thabo) cooks (food for Mpho) (**?** (41))
o				pheh	el	w		a	(Mpho) is being cooked (food) (**?** (42))

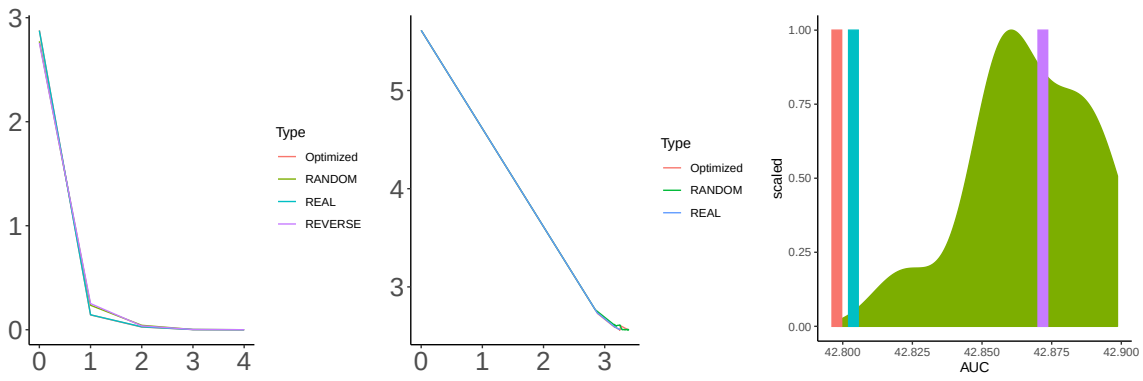


Figure 6: Japanese verb suffixes, measuring prediction on the level of morphemes, for real (blue), random (green), and approximately optimized (red) orderings. Left: I_t as a function of t. Center: Memory-surprisal tradeoff. Right: Areas under the curve for the memory-surprisal tradeoff.

	Pairs	Full
Optimized for Phoneme Prediction	0.976 (SD 0.007)	0.971 (SD 0.011)
Optimized for Morpheme Prediction	0.873 (SD 0.154)	0.85 (SD 0.184)
Random Baseline	0.519 (SD 0.177)	0.559 (SD 0.202)

Figure 7: Accuracy of approximately optimized orderings, and of random baseline orderings, in predicting verb suffix order in Japanese. 'Pairs' denotes the rate of pairs of morphemes that are ordered correctly, and 'Full' denotes the rate of verb forms where order is predicted entirely correctly. We show means and standard deviations over different runs of the optimization algorithm ('Optimized'), and over different random orderings ('Random').

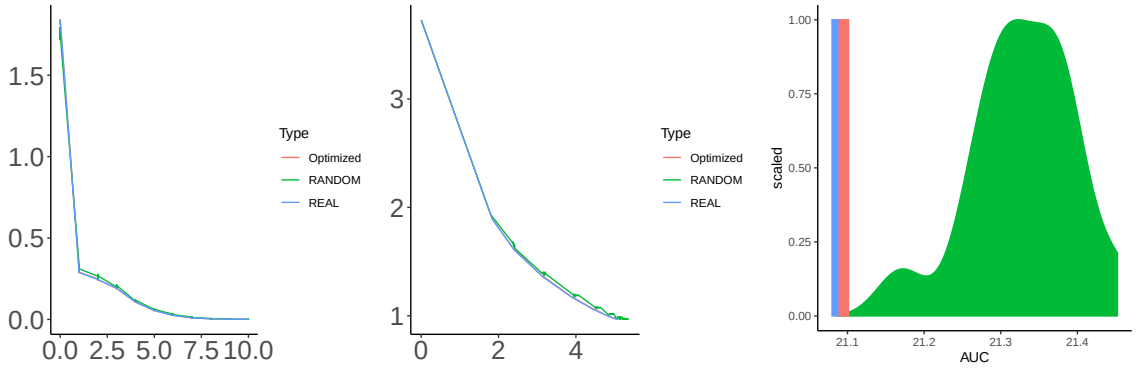


Figure 8: Sesotho verb affixes, measuring prediction on the level of phonemes, for real (blue), random (green), and approximately optimized (red) orderings. Left: I_t as a function of t. Center: Memory-surprisal tradeoff. Right: Areas under the curve for the memory-surprisal tradeoff.

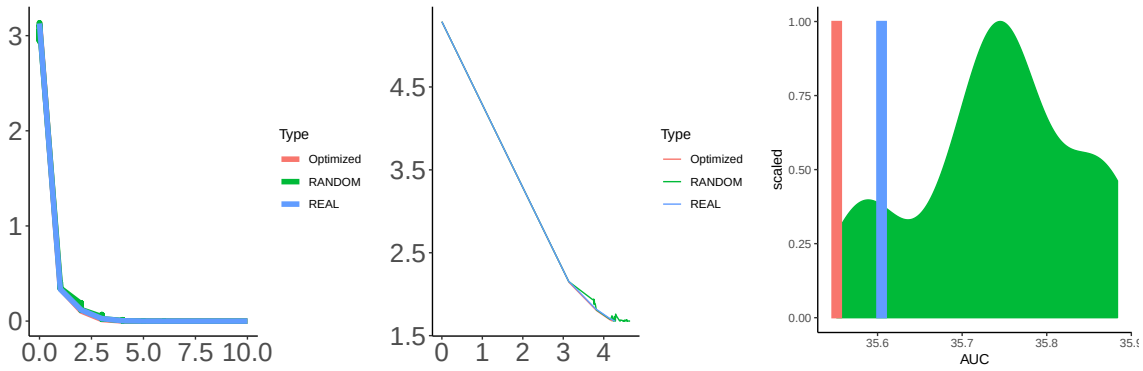


Figure 9: Sesotho verb affixes, measuring prediction on the level of morphemes, for real (blue), random (green), and approximately optimized (red) orderings. Left: I_t as a function of t. Center: Memory-surprisal tradeoff. Right: Areas under the curve for the memory-surprisal tradeoff.

		Prefixes		Suffixes	
		Pairs	Full	Pairs	Full
Phonemes	Optimized	0.987 (SD 0.004)	0.983 (SD 0.006)	0.914 (SD 0.131)	0.906 (SD 0.14)
	Random	0.468 (SD 0.321)	0.364 (SD 0.353)	0.459 (SD 0.224)	0.409 (SD 0.235)
Morphemes	Optimized	0.997 (SD 0.0)	0.995 (SD 0.0)	0.804 (SD 0.0)	0.721 (SD 0.0)
	Random	0.521 (SD 0.331)	0.43 (SD 0.349)	0.468 (SD 0.233)	0.401 (SD 0.251)

Figure 10: Accuracy of approximately optimized orderings, and of random baseline orderings, in predicting verb affix order in Sesotho. 'Pairs' denotes the rate of pairs of morphemes that are ordered correctly, and 'Full' denotes the rate of verb forms where order is predicted entirely correctly. We show means and standard deviations over different runs of the optimization algorithm ('Optimized'), and over different random orderings ('Random').

10.3 Results: Japanese

10.4 Results: Sesotho

11 Some theoretical thoughts (scratch area)

11.1 Learnability bounds

Upper bounds via n-gram models: Want to show that, if we know a process has stronger locality, there is a learning algorithm with lower sample complexity.

Theorem 5. *The class of processes with fixed bounds on H_0 on EE can be learned up to KL loss 2ε with ... samples with prob ...*

We want to learn a process to average KL distance 2ε. Assume excess entropy is $\leq I$, then only need to learn $N := I/\varepsilon$-gram-model for KL loss ε.

(Or, to get a better bound, take N so that $\sum_{t=N}^{\infty} I_t < \varepsilon$.).

So $N = \sum_{t=1}^{\infty} tI_t/\varepsilon$. How many samples are needed to learn this up to ε? Probably that will scale with the block entropy, which is $NH_0 + \sum_{t=1}^{N} tI_t$, with H_0 the unigram entropy. So the sample complexity would seem to scale with

$$\exp\left(H_0 \sum_{t=1}^{\infty} tI_t/\varepsilon + \sum_{t=1}^{\sum_{t=1}^{\infty} tI_t/\varepsilon} tI_t\right)$$

Want to learn $P(Y|X)$ up to $\mathbb{E}_x D_{KL}(P(y|x)||\hat{P}(y|x)) \leq \varepsilon$. How many i.i.d. samples from (X,Y) do we need?

Assume the distribution of Z has 'low' entropy. How many i.i.d. samples do we need to get a good estimate of $P(Z)$? Something scaling with $\exp(H(Z))$?

Convergence

references:

`https://arxiv.org/pdf/1904.02291.pdf` leads to:

$$Pr(D_{KL}(\hat{P}(z)||P(z)) \geq \varepsilon) \leq e^{-n\varepsilon}\left(\frac{e\varepsilon n}{|V|-1}\right)^{|V|-1}$$

if V is the set of values of Z, when $\varepsilon > \frac{|V|-1}{n}$.

Is it possible to get something similar but with $|V|$ replaced with $H[X] + |V_Y|$?

Theorem 6. *(must be something standard) Let X be an RV. Then $1-\varepsilon$ of its probability mass is concentrated on at most ??? values.*

Proof. Assume no way of covering $1-\varepsilon$ probability mass takes less than K values. That is (ordering p_i by magnitude downwards):

$$\sum_{i=1}^{K} p_i \leq 1-\varepsilon$$

Want to show that $H[X]$ cannot be too small.

First, note

$(1-\varepsilon) - (K-1)\frac{1-\varepsilon}{|V|-K} \geq p_1 \geq \cdots \geq p_K \geq \frac{1-\varepsilon}{|V|-K}$

or (reformulating slightly)

$(1-\varepsilon)(1-(K-1)\frac{1}{|V|-K}) \geq p_1 \geq \cdots \geq p_K \geq \frac{1-\varepsilon}{|V|-K}$

$H[X] \geq \sum_{i=1}^{K} \frac{1-\varepsilon}{|V|-K} \log \frac{1}{(1-\varepsilon)(1-\frac{K-1}{|V|-K})} = K \frac{1-\varepsilon}{|V|-K} \log \frac{1}{(1-\varepsilon)(1-\frac{K-1}{|V|-K})} = K \frac{1-\varepsilon}{|V|-K} \log \frac{1}{(1-\varepsilon)\frac{|V|-K-K+1}{|V|-K})} = K \frac{1-\varepsilon}{|V|-K} \log \frac{|V|-K}{(1-\varepsilon)(|V|-2K+1)}$

And so

$H[X] \geq K \frac{1-\varepsilon}{|V|-K} \log \frac{|V|-K}{(1-\varepsilon)(|V|-2K+1)} = (1-\varepsilon) \frac{K}{|V|-K} \log \frac{|V|-K}{(1-\varepsilon)(|V|-2K+1)}$ If $K << |V|$, this is close to zero.

—

So $H[X] \geq K \frac{1-\varepsilon}{|V|-K} \log \frac{|V|-K}{(1-\varepsilon)(|V|-K)} = \frac{K}{|V|-K}(1-\varepsilon) \log \frac{1}{(1-\varepsilon)}$

So

$\frac{K}{|V|-K} \leq H[X] \frac{1}{(1-\varepsilon) \log \frac{1}{(1-\varepsilon)}}$

However, this bound does not seem very useful, as it never allows one to conclude something like $K << |V|$.

Or, a better bound:

$H[X] \geq -(1-\varepsilon) \log \left[(1-\varepsilon) \left(\frac{|V|-2K+1}{|V|-K} \right) \right] + \varepsilon \log \frac{1}{\varepsilon}$

So If $K << |V|$, then $H[X] \geq ... \approx -(1-\varepsilon)\log(1-\varepsilon) + \varepsilon \log \frac{1}{\varepsilon}$.

□

To get intuition, if $I_t = \alpha \cdot t^{-3}$, then excess entropy $I = \alpha\pi^2/6$, and (bounds are a bit crude here)

$$H_0 \sum_{t=1}^{\infty} t\alpha t^{-3}/\varepsilon + \sum_{t=1}^{\sum_{t=1}^{\infty} t\alpha t^{-3}/\varepsilon} t\alpha \cdot t^{-3} \tag{55}$$

$$= H_0\alpha/\varepsilon \cdot \sum_{t=1}^{\infty} t^{1-3} + \sum_{t=1}^{\sum_{t=1}^{\infty} t\alpha t^{-3}/\varepsilon} t\alpha \cdot t^{-3} \tag{56}$$

$$\leq H_0\alpha/\varepsilon \cdot \pi^2/6 + \sum_{t=1}^{\alpha/\varepsilon \cdot \pi^2/6} t\alpha \cdot t^{-3} \tag{57}$$

$$= \frac{H_0\alpha\pi^2}{6\varepsilon} + \alpha \sum_{t=1}^{\alpha/\varepsilon \cdot \pi^2/6} t^{-2} \leq \frac{H_0\alpha\pi^2}{6\varepsilon} + \alpha\pi^2/6 \leq \left(\frac{H_0}{\varepsilon} + 1 \right) \frac{\alpha\pi^2}{6} \tag{58}$$

$$\tag{59}$$

Question: Can there be meaningful lower bounds in terms of locality?

Question: Can we give a lower bound by considering that learning sequences itself requires short-term memory?

try to derive something for Trading Value and Information paper

11.2 Optimization

Area under $t - H_t$ curve up to T: $\sum_{t=1}^{T} H_t = TH + \sum_{t=1}^{T} tI_t$

We cannot optimize for AUC using the method of (CITE), because we cannot construct an unbiased gradient estimator $AUC \propto \sum_t (\sum_{s \leq t} I_s) t I_t = \sum_{s \leq t} t I_t I_s = I_1^2 + 2I_1I_2 + 2I_2^2 + \ldots$

As a surrogate, we propose to maximize I_1. This corresponds

It provides an accurate approximation to the AUC if I_s is small for $s > 1$, which holds for the n-gram based estimator.

If $I_s < \varepsilon$ for $s > 1$, then

$$I_1^2 \leq \sum_{1 \leq s \leq t \leq T} t I_t I_s \leq I_1^2 + I_1 \varepsilon T^2 + \varepsilon^2 T^2 \tag{60}$$

That means, I_t^2 is an accurate approximation of the AUC when εT^2 is small.

Softmax gradient update corresponds to adding α to the target logit and removing α from all other logits. Equivalently, add a certain dynamic amount to the logcount of the target and the total logcount.

Counting corresponds to adding 1 to the target probability.