Predictive Querying for Autoregressive Neural Sequence Models

Alex Boyd[*1] **Sam Showalter**[*2] **Stephan Mandt**[1,2] **Padhraic Smyth**[1,2]
[1]Department of Statistics [2]Department of Computer Science
University of California, Irvine
{alexjb,showalte,mandt,p.smyth}@uci.edu

Abstract

In reasoning about sequential events it is natural to pose probabilistic queries such as "when will event A occur next" or "what is the probability of A occurring before B", with applications in areas such as user modeling, medicine, and finance. However, with machine learning shifting towards neural autoregressive models such as RNNs and transformers, probabilistic querying has been largely restricted to simple cases such as next-event prediction. This is in part due to the fact that future querying involves marginalization over large path spaces, which is not straightforward to do efficiently in such models. In this paper we introduce a general typology for predictive queries in neural autoregressive sequence models and show that such queries can be systematically represented by sets of elementary building blocks. We leverage this typology to develop new query estimation methods based on beam search, importance sampling, and hybrids. Across four large-scale sequence datasets from different application domains, as well as for the GPT-2 language model, we demonstrate the ability to make query answering tractable for arbitrary queries in exponentially-large predictive path-spaces, and find clear differences in cost-accuracy tradeoffs between search and sampling methods.

1 Introduction

One of the major successes in machine learning in recent years has been the development of neural sequence models for categorical sequences, particularly in natural language applications but also in other areas such as automatic code generation and program synthesis [Shin et al., 2019, Chen et al., 2021], computer security [Brown et al., 2018], recommender systems [Wu et al., 2017], genomics [Shin et al., 2021, Amin et al., 2021], and survival analysis [Lee et al., 2019]. Many of the models (although not all) rely on autoregressive training and prediction, allowing for the sequential generation of sequence completions in a recursive manner conditioned on sequence history.

A natural question in this context is how to compute answers to predictive queries that go beyond traditional one-step-ahead predictions. Examples of such queries are "how likely is event A to occur before event B?" and "how likely is event C to occur (once or more) within the next K steps of the sequence?" These types of queries are very natural across a wide variety of application contexts, for example, the probability that an individual will finish speaking or writing a sentence within the next K words, or that a user will use one app before another. See Fig. 1 and Appendix A for examples.

In this paper we develop a general framework for answering such predictive queries in the context of autoregressive (AR) neural sequence models. This amounts to computing conditional probabilities of propositional statements about future events, conditioned on the history of the sequence as summarized by the current hidden state representation. We focus in particular on how to perform near

[*]Authors contributed equally

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

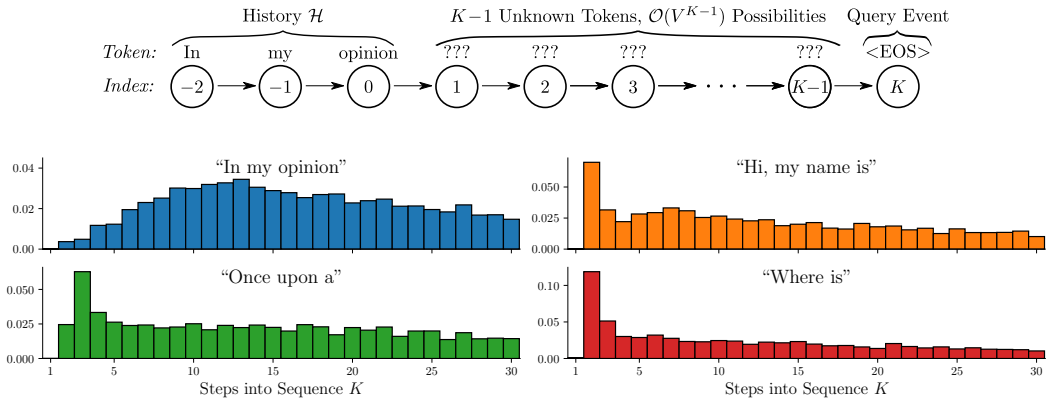

Figure 1: (top) Illustration of a query for the probability of a given sentence "In my opinion..." ending in K steps. (bottom) GPT-2 query estimates across 4 prefixes with $V = 50,257$, $K \leq 30$. Importance sampling query estimates maintain a 6x reduction in variance relative to naive model sampling for the same computation budget. Open-ended prefixes (top-left) generally possess longer-tailed distributions relative to simple prefixes. Additional details provided in Sections 4, 5, and Appendix H.5.

real-time computation of such queries, motivated by use-cases such as answering human-generated queries and utilizing query estimates within the optimization loop of training a model. Somewhat surprisingly, although there has been extensive prior work on multivariate probabilistic querying in areas such as graphical models and database querying, as well as for restricted types of queries for traditional sequence models such as Markov models, querying for neural sequence models appears to be unexplored. One possible reason is that the problem is computationally intractable in the general case (as we discuss later in Section 3), typically scaling as $O(V^{K-1})$ or worse for predictions K-steps ahead, given a sequence model with a V-ary alphabet (e.g. compared to $O(KV^2)$ for Markov chains).

Our contributions are three-fold:

1. We introduce and develop the problem of predictive querying in neural sequence models by reducing complex queries to elementary building blocks. These elementary queries define restricted path spaces over future event sequences.
2. We show that the underlying autoregressive model can always be constrained to the restricted path space satisfying a given query. This gives rise to a new proposal distribution that can be used for importance sampling, beam search, or a new hybrid approach.
3. We evaluate these methods across three user behavior and two language data datasets. While all three methods significantly improve over naive forward simulation, the hybrid approach further improves over importance sampling and beam search. We furthermore explore how the performance of all methods relates to the model entropy.

Code for this work is available at https://github.com/ajboyd2/prob_seq_queries.

2 Related Work

Research on efficient computation of probabilistic queries has a long history in machine learning and AI, going back to work on exact inference in multivariate graphical models [Pearl, 1988, Koller and Friedman, 2009]. Queries in this context are typically of two types. The first are *conditional probability queries*, the focus of our attention here: computing probabilities defined for a subset X of variables of interest, conditioned on a second subset $Y = y$ of observed variable values, and marginalizing over the set Z of all other variables. The second type of queries can broadly be referred to as *assignment queries*, seeking the most likely (highest conditional probability) assignment of values x for X, again conditioned on $Y = y$ and marginalizing over the set Z. Assignment queries are also referred to as most probable explanation (MPE) queries, or as maximum a posteriori (MAP) queries when Z is the empty set [Koller et al., 2007].

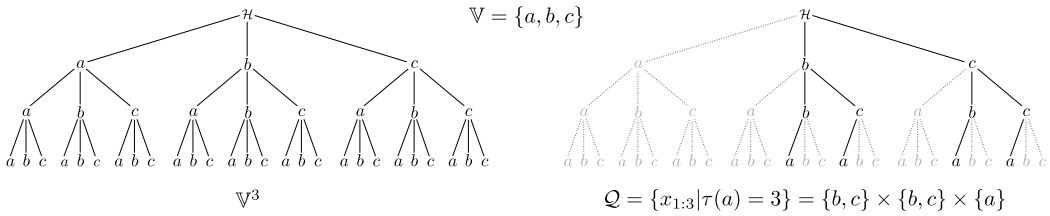

Figure 2: (left) Tree diagram of the complete sequence space for a vocabulary $\mathbb{V} = \{a, b, c\}$ and the corresponding query space $\mathcal{Q}$ (right) for when *the first appearance of* a occurs on the third step (i.e., $\tau(a) = 3$), defined as the set product of restricted domains listed below the figure.

For models that can be characterized with sparse Markov dependence structure, there is a significant body of work on efficient inference algorithms that can leverage such structure [Koller and Friedman, 2009], in particular for sequential models where recursive computation can be effectively leveraged [Bilmes, 2010]. However, autoregressive neural sequence models are inherently non-Markov since the real-valued current hidden state is a function of the entire history of the sequence. Each hidden state vector induces a tree containing V^k unique future trajectories with state-dependent probabilities for each path. Techniques such as dynamic programming (used effectively in Markov-structured sequence models) are not applicable in this context, and both assignment queries and conditional probability queries are NP-hard in general Chen et al. [2018].

For assignment-type queries there has been considerable work in natural language processing with neural sequence models, particularly for the MAP problem of generating high-quality/high-probability sequences conditioned on sequence history or other conditioning information. A variety of heuristic decoding methods have been developed and found to be useful in practice, including beam search [Sutskever et al., 2014], best-first search [Xu and Durrett, 2021], sampling methods [Holtzman et al., 2019], and hybrid variants [Shaham and Levy, 2022]. However, for conditional probability queries with neural sequence models (the focus of this paper), there has been no prior work in general on this problem to our knowledge. While decoding techniques such as beam search can also be useful in the context of conditional probability queries, as we will see later in Section 4, such techniques have significant limitations in this context, since by definition they produce lower-bounds on the probabilities of interest and, hence, are biased estimators.

3 Probabilistic Queries

Notation Let $X_{1:N} := [X_1, X_2, \ldots, X_N]$ be a sequence of random variables with arbitrary length N. Additionally, let $x_{1:N} := [x_1, x_2, \ldots, x_N]$ be their respective observed values where each x_i takes on values from a fixed vocabulary $\mathbb{V} := \{1, \ldots, V\}$. Examples of these sequences include sentences where each letter or word is a single value, or streams of discrete events generated by some process or user. We will refer to individual variable-value pairs in the sequence as events.

We consider an autoregressive model $p_\theta(X_{1:N}) = \prod_{i=1}^{N} p_\theta(X_i | X_{1:i-1})$, parameterized by θ and trained on a given dataset of M independent draws from a ground truth distribution $\mathbb{P}(X_{1:N})$. We assume that this model can be conditioned on a subsequence, termed the *history* $\mathcal{H}$. We will remap the indices of subsequent random variables to start at 1[2]. We abbreviate conditioning on a history by an asterisk *, i.e., $\mathbb{P}^*(\cdot) := \mathbb{P}(\cdot|\mathcal{H})$ and $p^*_\theta(\cdot) := p_\theta(\cdot|\mathcal{H})$.

Defining Probabilistic Queries Given a specific history of events $\mathcal{H}$, there are a variety of different questions one could ask about the future continuing where the history left off: (Q1) What event is likely to happen next? (Q2) Which events are likely to occur $k > 1$ steps from now? (Q3) What is the distribution of when the next instance of $a \in \mathbb{V}$ occurs? (Q4) How likely is it that we will see event $a \in \mathbb{V}$ occur before $b \in \mathbb{V}$? (Q5) How likely is it for $a \in \mathbb{V}$ to occur n times in the next k steps?

We define a common framework for such queries by defining probabilistic queries to be of the form $p^*_\theta(X_{1:K} \in \mathcal{Q})$ with $\mathcal{Q} \subset \mathbb{V}^K$. This can be extended to the infinite setting (i.e., $p^*_\theta([X_k]_k \in \mathcal{Q})$

[2]For example, if $|\mathcal{H}| = 3$ then $\mathbb{P}(X_4|\mathcal{H})$ is distribution of the 7th value in a sequence after conditioning on the first 3 values in the sequence.

#	Question	Probabilistic Query	$K \cdot \lvert\mathcal{Q}\rvert$
Q1	Next event?	$p^*_\theta(X_1)$	$\mathcal{O}(1)$
Q2	Event K steps from now?	$p^*_\theta(X_K)$	$\mathcal{O}(V^{K-1})$
Q3	Next instance of a?	$p^*_\theta(\tau(a) = K)$	$\mathcal{O}((V-1)^{K-1})$
Q4	Will a happen before b?	$p^*_\theta(\tau(a) < \tau(b))$	$\mathcal{O}((V-2)^{K})^\dagger$
Q5	How many instances of a in K steps?	$p^*_\theta(N_a(K) = n)$	$\mathcal{O}(\binom{K}{n}(V-1)^{K-n})$

Table 1: List of example questions, corresponding probabilistic queries, and associated costs of exact computation computation with an autoregressive model. The cost of accommodating a history $\mathcal{H}$ is assumed to be an additive constant for all queries. While Q4 extends to infinite time, the cost reported is for computing a lower bound up to K steps.

where $\mathcal{Q} \subset \mathbb{V}^\infty$). Exact computation of an arbitrary query is straightforward to represent:

$$p^*_\theta(X_{1:K} \in \mathcal{Q}) = \sum_{x_{1:K} \in \mathcal{Q}} p^*_\theta(X_{1:K} = x_{1:K}) = \sum_{x_{1:K} \in \mathcal{Q}} \prod_{k=1}^{K} p^*_\theta(X_k = x_k | X_{<k} = x_{<k}). \quad (1)$$

Depending on $|\mathcal{Q}|$, performing this calculation can quickly become intractable, motivating lower bounds or approximations (developed in more detail in Section 4). In this context it is helpful to impose structure on the query $\mathcal{Q}$ to make subsequent estimation easier, in particular by breaking $\mathcal{Q}$ into the following structured partition:

$$\mathcal{Q} = \cup_i \mathcal{Q}^{(i)} \text{ where } \mathcal{Q}^{(i)} \cap \mathcal{Q}^{(j)} = \emptyset \text{ for } i \neq j \quad (2)$$

$$\text{and } \mathcal{Q}^{(i)} = \mathcal{V}_1^{(i)} \times \mathcal{V}_2^{(i)} \times \cdots \times \mathcal{V}_K^{(i)} \text{ where } \mathcal{V}_k^{(i)} \subseteq \mathbb{V} \text{ for } k = 1, \ldots, K. \quad (3)$$

In words, this means a given query $\mathcal{Q}$ can be broken into a partition of simpler queries $\mathcal{Q}^{(i)}$ which take the form of a set cross product between restricted domains $\mathcal{V}_k^{(i)}$, one domain for each token X_k.[3] An illustration of an example query set can be seen in Fig. 2. A natural consequence of this is that:

$$p^*_\theta(X_{1:K} \in \mathcal{Q}) = \sum_i p^*_\theta(X_{1:K} \in \mathcal{Q}^{(i)}) = \sum_i p^*_\theta(\cap_{k=1}^K X_k \in \mathcal{V}_k^{(i)}),$$

which lends itself to more easily estimating each term in the sum. This will be discussed in Section 4.

Queries of Interest All of the queries posed earlier in this section can be represented under the framework detailed in Eq. (2) and Eq. (3), as illustrated in Table 1.

Q1 & Q2 The queries $p^*_\theta(X_1 = a)$ and $p^*_\theta(X_K = a)$ for some $a \in \mathbb{V}$ can be represented with $\mathcal{Q} = \{a\}$ and $\mathcal{Q} = \mathbb{V}^{K-1} \times \{a\}$ respectively.

Q3 The probability of the next instance of $a \in \mathbb{V}$ occurring at some point in time $K \geq 1$, $p^*_\theta(\tau(a) = K)$ where $\tau(\cdot)$ is the *hitting time*, can be represented as $\mathcal{Q} = (\mathbb{V} \setminus \{a\})^{K-1} \times \{a\}$. This can be adapted for a set $A \subset \mathbb{V}$ by replacing $\{a\}$ in $\mathcal{Q}$ with A.

Q4 The probability of $a \in \mathbb{V}$ occurring before $b \in \mathbb{V}$, $p^*_\theta(\tau(a) < \tau(b))$, is represented as $\mathcal{Q} = \cup_{i=1}^\infty \mathcal{Q}^{(i)}$ where $\mathcal{Q}^{(i)} = (\mathbb{V} \setminus \{a, b\})^{i-1} \times \{a\}$. Lower bounds to this can be computed by limiting $i < i'$. Like Q3, this can also be extended to disjoint sets $A, B \subset \mathbb{V}$.

Q5 The probability of $a \in \mathbb{V}$ occurring n times in the next K steps, $p^*_\theta(N_a(K) = n)$, is represented as $\mathcal{Q} = \cup_{i=1}^{C(K,n)} \mathcal{Q}^{(i)}$, where $N_a(K)$ is a random variable for the number of occurrences of events of type a from steps 1 to K and $\mathcal{Q}^{(i)}$'s are defined to cover all unique permutations of orders of products composed of: $\{a\}^n$ and $(\mathbb{V} \setminus \{a\})^{K-n}$. Like above, this can easily be extended for $A \subset \mathbb{V}$.

Query Complexity From Eq. (1), exact computation of a query involves computing $K \cdot |Q|$ conditional distributions (e.g., $p^*_\theta(X_k | X_{<k} = x_{<k})$) in an autoregressive manner. Under the structured representation, the number of conditional distributions needed is equivalently $\sum_i \prod_{k=1}^K |\mathcal{V}_k^{(i)}|$. Non-attention based neural sequence models often define $p^*_\theta(X_k | X_{<k} = x_{<k}) := f_\theta(h_k)$ where

[3] Ideally, the partitioning is chosen to have the smallest number of $\mathcal{Q}^{(i)}$'s needed.

$h_k = \text{RNN}_\theta(h_{k-1}, x_{k-1})$. As such, the computation complexity for any individual conditional distribution remains constant with respect to sequence length. We will refer to the complexity of this atomic action as being $\mathcal{O}(1)$. Naturally, the actual complexity depends on the model architecture and has a multiplicative scaling on the cost of computing a query. The number of atomic operations needed to exactly compute Q1-Q5 for this class of models can be found in Table 1. Should p_θ be an attention based model (e.g., a transformer [Vaswani et al., 2017]) then the time complexity of computing a single one-step-ahead distribution becomes $\mathcal{O}(K)$, further exacerbating the **exponential growth** of many queries. Note that with some particular parametric forms of p_θ, query computation can be more efficient, e.g., see Appendix C for a discussion on query complexity for Markov models.

4 Query Estimation Methods

Since exact query computation can scale exponentially in K it is natural to consider approximation methods. In particular we focus on importance sampling, beam search, and a hybrid of both methods. All methods will be based on a novel proposal distribution, discussed below.

4.1 Proposal Distribution

For various estimation methods which will be discussed later, it is beneficial to have a proposal distribution $q(X_{1:K} = x_{1:K})$ whose domain matches that of the query $\mathcal{Q}$. For importance sampling, we will need this distribution as a proposal distribution, while we use it as our base model for selecting high-probability sequences in beam search. We would like the proposal distribution to resemble our original model while also respecting the query. One thought is to have $q(X_{1:K} = x_{1:K}) = p^*_\theta(X_{1:K} = x_{1:K} | X_{1:K} \in \mathcal{Q})$. However, computing this probability involves normalizing over $p^*_\theta(X_{1:K} \in \mathcal{Q})$ which is exactly what we are trying to estimate in the first place. Instead of restricting the *joint* distribution to the query, we can instead restrict every *conditional* distribution to the query's restricted domain. To see this, we first partition $\mathcal{Q} = \cup_i \mathcal{Q}^{(i)}$ and define an autoregressive proposal distribution for each $\mathcal{Q}^{(i)} = \prod_{k=1}^{K} \mathcal{V}_k^{(i)}$ as follows:

$$
\begin{aligned}
q^{(i)}(X_{1:K} = x_{1:K}) &= \prod_{k=1}^{K} p^*_\theta(X_k = x_k | X_{<k} = x_{<k}, X_k \in \mathcal{V}_k^{(i)}) \\
&= \prod_{k=1}^{K} \frac{p^*_\theta(X_k = x_k | X_{<k} = x_{<k}) \mathbb{1}(x_k \in \mathcal{V}_k^{(i)})}{\sum_{v \in \mathcal{V}_k^{(i)}} p^*_\theta(X_k = v | X_{<k} = x_{<k})}
\end{aligned}
\tag{4}
$$

where $\mathbb{1}(\cdot)$ is the indicator function. That is, we constrain the outcomes of each conditional probability to the restricted domains $\mathcal{V}_k^{(i)}$ and renormalize them accordingly. To evaluate the proposal distribution's probability, we multiply all conditional probabilities according to the chain rule. Since the entire distribution is computed for a single model call $p^*_\theta(X_k | X_{<k} = x_{<k})$, it is possible to both sample a K-length sequence and compute its likelihood under $q^{(i)}$ with only K model calls. Thus, we can efficiently sample sequences from a distribution that is both informed by the underlying model p_θ and that respects the given domain $\mathcal{Q}$. As discussed in the next section, this proposal will be used for importance sampling and for the base distribution on which beam search is conducted.

4.2 Estimation Techniques

Sampling One can naively sample any arbitrary probability value using Monte Carlo samples to estimate $p^*_\theta(X_{1:K} \in \mathcal{Q}) = \mathbb{E}_{p^*_\theta}[\mathbb{1}(X_{1:K} \in \mathcal{Q})]$; however, this typically will have high variance. This can be substantially improved upon by exclusively drawing sequences from the query space $\mathcal{Q}$. Arbitrary queries can be written as a sum of probabilities of individual sequences, as seen in Eq. (1). This summation can be equivalently written as an expected value,

$$
p^*_\theta(X_{1:K} \in \mathcal{Q}) = \sum_{x_{1:K} \in \mathcal{Q}} p^*_\theta(X_{1:K} = x_{1:K}) = |\mathcal{Q}| \, \mathbb{E}_{x_{1:K} \sim \mathcal{U}(\mathcal{Q})} \left[p^*_\theta(X_{1:K} = x_{1:K}) \right],
$$

where $\mathcal{U}$ is a uniform distribution. It is common for p^*_θ to concentrate most of the available probability mass on a small subset of the total possible space $\mathbb{V}^K$. Should $|\mathcal{Q}|$ be large, then $|\mathcal{Q}| \; p^*_\theta(X_{1:K} = x_{1:K})$

will have very large variance for $x_{1:K} \sim \mathcal{U}(\mathcal{Q})$. One way to mitigate this is to use importance sampling with our proposal distribution q (Section 4.1):

$$p^*_\theta(X_{1:K} \in \mathcal{Q}) = |\mathcal{Q}| \, \mathbb{E}_{x_{1:K} \sim \mathcal{U}(\mathcal{Q})} \left[p^*_\theta(X_{1:K} = x_{1:K}) \right] = \mathbb{E}_{x_{1:K} \sim q} \left[\frac{p^*_\theta(X_{1:K} = x_{1:K})}{q(X_{1:K} = x_{1:K})} \right]$$
$$\approx \frac{1}{M} \sum_{m=1}^{M} \frac{p^*_\theta(X_{1:K} = x^{(m)}_{1:K})}{q(X_{1:K} = x^{(m)}_{1:K})} \qquad \text{for } x^{(1)}_{1:K}, \ldots, x^{(M)}_{1:K} \overset{iid}{\sim} q.$$

It is worth noting that this estimator could be further improved by augmenting the sampling process to produce samples without replacement from q (e.g., [Meister et al., 2021, Kool et al., 2019, Shi et al., 2020]); for the purposes of the scope of this paper we focus on sampling with replacement.

Search An alternative to estimating a query by sampling is to instead produce a lower bound,

$$p^*_\theta(X_{1:K} \in \mathcal{Q}) = \sum_{x_{1:K} \in \mathcal{Q}} p^*_\theta(X_{1:K} = x_{1:K}) \geq \sum_{x_{1:K} \in \mathcal{B}} p^*_\theta(X_{1:K} = x_{1:K}),$$

where $\mathcal{B} \subset \mathcal{Q}$. In many situations, only a small subset of sequences $x_{1:K}$ in $\mathcal{Q}$ have a non-negligible probability of occurring due to the vastness of the total path space V^K. As such, it is possible for $|\mathcal{B}| \ll |\mathcal{Q}|$ while still having a minimal gap between the lower bound and the actual query value.

One way to produce a set $\mathcal{B} \subset \mathcal{Q}$ is through beam search. To ensure that beam search only explores the query space, instead of searching with p_θ we utilize q for ranking beams. Since beam search is a greedy algorithm and for a given conditional $q(X_k = a | X_{<k}) \propto p^*_\theta(X_k = a | X_{<k})$ for $a \in \mathcal{V}_k$, the rankings will both respect the domain and be otherwise identical to using p_θ to rank. Typically, the goal of beam search is to find the most likely completion of a sequence without having to explore the entire space of possible sequences. This is accomplished by greedily selecting the top-B most likely next step continuations, or *beams*, at each step into the future. Rather than finding a few high-likelihood beams, we are more interested in accumulating a significant amount of probability and less so in the specific quantity of beams collected.

Traditional beam search has a fixed beam size B that is considered for its search; however, this is not ideal for collecting probability mass. As an alternative we develop *coverage-based* beam search where at each step in a sequence we restrict the set of beams being considered not to the top-B but rather to the smallest set of beams that collectively exceed a predetermined probability mass α, referred to as the "coverage".[4] More specifically, let $\mathcal{B}_k \subset \{x_{1:k} | x_{1:K} \in \mathcal{Q}\}$ be a set containing $|\mathcal{B}_k|$ beams for subsequences of length k. For brevity, we will assume that $\mathcal{Q} = \mathcal{V}_1 \times \cdots \times \mathcal{V}_K$.[5] $\mathcal{B}_{k+1}$ is a subset of $\mathcal{B}_k \times \mathcal{V}_{k+1}$ and is selected specifically to minimize $|\mathcal{B}_{k+1}|$ such that $q(X_{1:k+1} \in \mathcal{B}_{k+1}) \geq \alpha$. It can be shown that $p^*_\theta(X_{1:K} \in \mathcal{Q}) - p^*_\theta(X_{1:K} \in \mathcal{B}_K) \leq 1 - q(X_{1:K} \in \mathcal{B}_K)$ (and is often quite less). See Appendix F for a proof.

There is one slight problem with having α be constant throughout the search. Since we are pruning based on the joint probability of the entire sequence, any further continuations of $\mathcal{B}_k$ will reduce the probability $q(X_{1:k+1} \in \mathbb{B}_{k+1})$. This may lead to situations in which every possible candidate sequence is kept in order to maintain minimal joint probability coverage. This can be avoided by filtering by α_k where $\alpha_1 > \cdots > \alpha_K = \alpha$, e.g., the geometric series $\alpha_k = \alpha^{k/K}$.

A Hybrid Approach Importance sampling produces an unbiased estimate, but can still experience large variance in spite of a good proposal distribution q when p_θ is a heavy tailed distribution. Conversely, the beam search lower bound can be seen as a biased estimate with zero variance. We can remedy the limitations of both methods by recognizing that since $p^*_\theta(X_{1:K} \in \mathcal{Q}) = \sum_{x_{1:K} \in \mathcal{B}_K} p^*_\theta(X_{1:K} = x_{1:K}) + \sum_{x_{1:K} \in \mathcal{Q} \setminus \mathcal{B}_K} p^*_\theta(X_{1:K} = x_{1:K})$ where $\mathcal{B}_K$ is the set of sequences resulting from beam search, we can use importance sampling on the latter summation. The only caveat to this is that the proposal distribution must match the same domain of the summation: $\mathcal{Q} \setminus \mathcal{B}_K$.

The proposal distribution we use to accomplish this is $q(X_{1:K} = x_{1:K} | X_{1:K} \notin \mathcal{B}_K)$ (see Eq. 4). This is implemented by storing all intermediate distributions (both original p_θ and proposal q) found

[4] This is similar to the distinction between top-K and top-p / nucleus sampling commonly used for natural language generation [Holtzman et al., 2019].

[5] If $\mathcal{Q}$ requires partitioning into multiple $\mathcal{Q}^{(i)}$'s, we apply beam search to each sub query $p^*_\theta(X_{1:K} \in \mathcal{Q}^{(i)})$.

during beam search and arranging them into a tree structure. All leaf nodes that are associated with $x_{1:K} \in \mathcal{B}_K$ have their transition probability zeroed out and then the effect of restriction and normalization is propagated up the tree to their ancestor nodes (details provided in Appendix D and E). Sampling from this new proposal distribution involves sampling from this normalized tree until we reach either an existing K-depth leaf node, in which case we are done, or a $< K$-depth leaf node in which case we complete the rest of the sequence by sampling from q in Eq. (4) as usual.

Lastly, since our ultimate goal is to sample from the long tail of p_θ, targeting a specific coverage α during beam search is no longer effective since achieving meaningfully large coverage bounds for non-trivial path spaces is generally intractable. Instead, we propose *tail-splitting* beam search to better match our goals. Let $w_k^{(i)} = p_\theta^*(X_{1:k+1} = x_{1:k+1}^{(i)})$ for $x_{1:k+1}^{(i)} \in \mathcal{B}_k \times \mathcal{V}_{k+1}$ such that $w_k^{(i)} \geq w_k^{(j)}$ if $i < j$. In this regime, $\mathcal{B}_{k+1} = \{x_{1:k+1}^{(i)}\}_{i=1}^B$ where $B = \arg\min_b \sigma(w_k^{(1:b)}) + \sigma(w_k^{(b+1:|\mathcal{B}_k \times \mathcal{V}_{k+1}|)})$. $\sigma(w_k^{(u:v)})$ is the empirical variance of $w_k^{(i)}$ for $i = u, \ldots, v$. This can be seen as performing 2-means clustering on the w_k's and taking the cluster with the higher cumulative probability.

4.3 Saving Computation on Multiple Queries

Should multiple queries need to be performed, such as $p_\theta^*(\tau(a) = k)$ for multiple values of k, then there is potential to be more efficient in computing estimates for all of them. The feasibility of reusing intermediate computations is dependent on the set of queries considered. For simplicity, we will consider two base queries $\mathcal{Q} = \mathcal{V}_1 \times \cdots \times \mathcal{V}_K$ and $\mathcal{Q}' = \mathcal{V}_1' \times \cdots \times \mathcal{V}_{K'}'$ where $K < K'$. Due to the autoregressive nature of p_θ, if $\mathcal{V}_i = \mathcal{V}_i'$ for $i = 1, \ldots, K-1$ then all of the distributions and sequences needed for estimating $p_\theta^*(X_{1:K} \in \mathcal{Q})$ are guaranteed to be intermediate results found when estimating $p_\theta^*(X_{1:K'} \in \mathcal{Q}')$. To be explicit, when estimating the latter query with beam search the intermediate $\mathcal{B}_K$ is the same as what would be directly computed for the former query. Likewise, for importance sampling if $x_{1:K'} \sim q(X_{1:K'})$ using Eq. (4) then the subsequence $x_{1:K} \sim q(X_{1:K})$. This does not apply when the sample path domain is further restricted, such as with the hybrid approach, in which case we cannot directly use intermediate results to compute other queries for "free."

5 Experiments and Results

Experimental Setting We investigate estimates of hitting time queries across various datasets, comparing beam search, importance sampling, and the hybrid method. We find that hybrid systematically outperforms both pure search and sampling given a comparable computation budget across queries and datasets. We also investigate the dependence of all three methods on the model entropy.

It is worth noting that we focus almost exclusively on hitting time queries in our primary experiments as more complex queries *often decompose into operations over individual hitting times*. For example. consider the following decomposition of the "A before B" query (Q4):

$$p_\theta^*(\tau(a) < \tau(b)) = \sum_{k=1}^{\infty} p_\theta^*(\tau(a) = k, \tau(b) > k) = \sum_{k=1}^{\infty} p_\theta^*(X_k = a, X_{<k} \in (\mathbb{V} \setminus \{a, b\})^{k-1})$$

Decompositions of other queries and their impact on approximation error are found in Appendix B.

Datasets We evaluate our query estimation methods on three user behavior and two language sequence datasets. **Reviews** [Ni et al., 2019] contains sequences of Amazon customer reviews for products belonging to one of $V = 29$ categories; **Mobile Apps** [Aliannejadi et al., 2021] consists of app usage records over $V = 88$ unique applications; **MOOCs** consists of student interaction with online course resources over $V = 98$ actions. We also use the works of William **Shakespeare** [Shakespeare] by modeling the occurrence of $V = 67$ unique ASCII characters. Lastly, we examine **WikiText** [Merity et al., 2017] to explore word-level sequence modeling applications with GPT-2, a large-scale language model with a vocabulary of $V = 50257$ word-pieces [Radford et al., 2019, Wu et al., 2016]. After preprocessing, no datasets contain personal identifiable information.

Base Models While our proposed methods are amenable to autoregressive models of arbitrary structure, we focus our analysis specifically on recurrent neural networks. For all datasets except WikiText, we train Long-short Term Memory (LSTM) networks until convergence. To explore

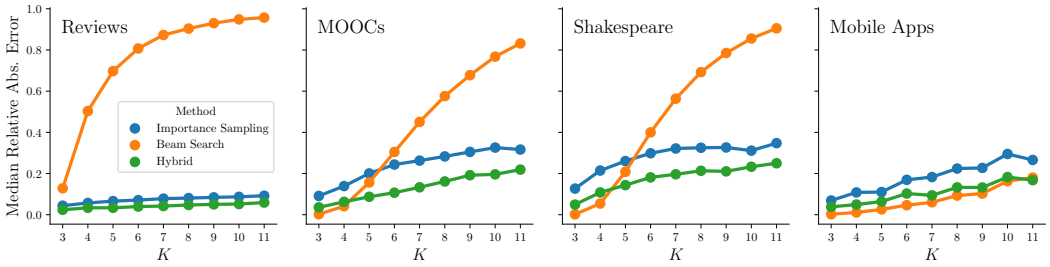

Figure 3: Median relative absolute error (RAE) between estimated probability and (surrogate) ground truth for $p^*_\theta(\tau(\cdot) = K)$ for importance sampling, beam search, and the hybrid method. As query path space grows with K, beam search quickly fails to bound ground truth while sampling remains robust, with the hybrid consistently outperforming all other methods, especially for large values of K. Ground truth values used to determine error are exact for $K \leq 4$ and approximated otherwise.

modern applications of sequence modeling with WikiText, we utilize GPT-2 with pre-trained weights from HuggingFace [Wolf et al., 2020]. Model training and experimentation utilized roughly 200 NVIDIA GeForce 2080ti GPU hours. Please refer to Appendix G for additional details.

Experimental Methods We investigate computation-accuracy trade-offs between 3 estimation methods (beam search, importance sampling, and the hybrid) across all datasets. Query histories $\mathcal{H}$ are defined by randomly sampling $N = 1000$ sequences from the test split for all datasets except WikiText, from which we sample only $N = 100$ sequences due to computational limitations. For each query history and method, we compute the hitting time query estimate $p^*_\theta(\tau(a) = K)$ over $K = 3, \ldots, 11$, with a determined by the Kth symbol of the ground truth sequence.

To ensure an even comparison of query estimators, we fix the computation budget per query in terms of model calls $f_\theta(h_k)$ to be equal across all 3 methods, repeating experiments for different budget magnitudes roughly corresponding to $O(10), O(10^2), O(10^3)$ model calls (see Appendix H for full details). We intentionally select relatively small computation budgets per query to support systematic large-scale experiments over multiple queries up to relatively large values of K. Results for queries with GPT-2 are further restricted because of computational limits and are reported separately below.

To evaluate the accuracy of the estimates for each query and method, we compute the true probability of K using exact computation for small values of $K \leq 4$. For larger values of K, we run importance sampling with a large number of samples S, where S is adapted per query to ensure the resulting unbiased estimate has an empirical variance less than $\epsilon \ll 1$ (see Appendix G.3). This computationally-expensive estimate is then used as a surrogate for ground truth in error calculations.

Coverage-based beam search is not included in our results: we found that it experiences exponential growth with respect to K and does not scale efficiently due to its probability coverage guarantees. Additional details are provided in Appendix H.3.

Results: Accuracy and Query Horizon Using the methodology described above, for each query, we compute the relative absolute error (RAE) $|p - \hat{p}|/p$, where $\hat{p}$ is the estimated query probability generated by a particular method and p is the ground truth probability or the surrogate estimate using importance sampling. For each dataset, for each of the 3 levels of computation budget, for each value of K, this yields $N = 1000$ errors for the N queries for each method.

Fig. 3 shows the median RAE of the N queries, per method, as a function of K, for each dataset, using the medium computation budget in terms of model calls. Across the 4 datasets the error increases systematically as K increases. However, beam search is significantly less robust than the other methods for 3 of the 4 datasets: the error rate increases rapidly in a non-linear fashion compared to importance sampling and hybrid. Beam search is also the most variable across datasets relative to other methods. The hybrid method systematically outperforms importance sampling along across all 4 datasets and for all values of K. In Appendix H we provide additional results; for the lowest and highest levels of computational budget, for mean (instead of median) RAEs, and scatter plots for specific values of K with more detailed error information. The qualitative conclusions are consistent across all plots.

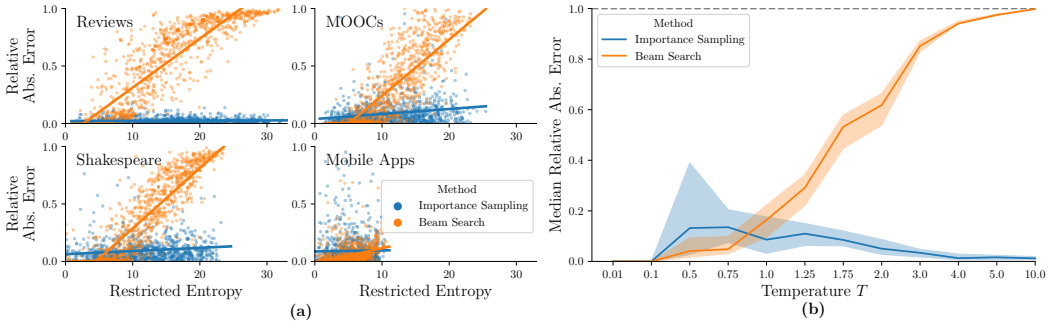

Figure 4: (left) RAE vs restricted entropy per query (with best linear fits), (right) Median RAE versus model temperature T for Mobile App data. All errors computed using the same queries as in Fig. 3. Beam search errors correlate highly with model entropy even with the low-entropy Mobile Apps dataset, where increasing temperature T directly induces this failure mode.

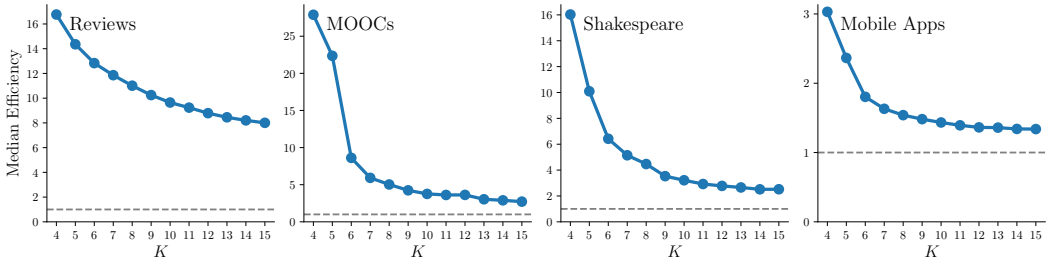

Figure 5: Median relative efficiency (over 1000 query histories and all vocabulary terms) of importance sampling estimation of the K-step marginal distributions for each dataset. The gray, dotted line represents 100% relative efficiency defined by naive query estimation. Relative efficiency is documented for $4 \leq K \leq 15$ to highlight the regime where ground truth cannot be tractably computed.

Results: The Effect of Model Entropy We conjecture that the entropy of the proposal distribution q conditioned on a given history $H_q^*(X_{1:K}) = -\mathbb{E}_{x_{1:K}\sim q}[\log q(X_{1:K} = x_{1:K})]$, which we refer to as *restricted entropy*, is a major factor in the performance of the estimation methods. Fig. 4(a) shows the RAE per query (with a linear fit) as a function of estimated restricted entropy for importance sampling and beam search. The results clearly show that entropy is driving the query error in general and that the performance of beam search is much more sensitive to entropy than sampling. The difference in entropy characteristics across datasets explains the differences in errors we see in Fig. 3. In particular, the Mobile Apps dataset is in a much lower entropy regime than the other 3 datasets.

To further investigate the effect of entropy, we alter each model by applying a temperature $T > 0$ to every conditional factor: $p_{\theta,T}(X_k|X_{<k}) \propto p_\theta(X_k|X_{<k})^{1/T}$, effectively changing the entropy ranges for the models. Fig. 4(b) shows the median RAE, for query $p^*_{\theta,T}(\tau(\cdot) = 4)$, as a function of model temperature for the Mobile Apps data. As predicted from Fig. 4(a), the increase in T, and corresponding increase in entropy, causes beam search's error to converge to 1, while the sampling error goes to 0. As T increases, each individual sequence will approach having $1/|\mathcal{Q}|$ mass, thus needing many more beams to have adequate coverage. Results for other queries and the other 3 datasets (in Appendix H) further confirm the fundamental bifurcation of error between search and sampling (that we see in Fig. 4(b)) as a function of entropy.

Results: Relative Efficiency of Proposal Distribution over Naive Query Estimation We also examine the relative efficiency improvements of our proposal distribution against naive Monte Carlo sampling:

$$p^*_\theta(X_{1:K} \in \mathcal{Q}) = \mathbb{E}_{x_{1:K}\sim p^*}[\mathbb{1}(x_{1:K} \in \mathcal{Q})]$$

Our relative efficiency calculations in Fig. 5 represent the variance ratio of naive query estimates and estimates from our query proposal distribution. As shown, all datasets witness improvement over

naive sampling efficiency and often by a significant margin. We also observe that relative efficiency is largest for shorter query horizons, approaching naive sampling efficiency as K increases.

Results: Queries with a Large Language Model We further explore entropy's effect on query estimation with GPT-2 and the WikiText dataset for $N = 100$, $K = 3, 4$, across 3 computation budgets. With a vocabulary 500x larger than the other models, GPT-2 allows us to examine queries relevant to NLP applications. The resulting high entropy causes beam search to fail to match surrogate ground truth given the computation budgets (consistent with earlier experiments), with a median RAE of 82% (for $K = 4$ and a budget of $O(10^3)$). By contrast, importance sampling's median RAE under the same setting is 13%, **a 6x reduction**. Additional results are in Appendix H.5.

6 Discussion

Future Directions: This work provides a starting point for multiple different research directions beyond the scope of this paper. One such direction is exploring more powerful search and sampling methods for query estimation. This includes methods for sampling without replacement, such as the Gumbel-max method [Huijben et al., 2022, to appear], sampling importance-resampling (SIR) [Skare et al., 2003], as well as new heuristics for automatically trading off search and sampling on a per-query basis. Another direction for exploration is amortized methods, such as learning models before queries are issued, that are specifically designed to help answer queries. Learning models that include queries as part of regularization terms in objective functions can also build on this work, e.g., learning models that don't only rely on one-step-ahead losses [Meister and Cotterell, 2021]. The work in this paper could also be broadened to continuous-time models for marked temporal point processes (requiring marginalization over time as well as event types).

Limitations: Our results rely on only four datasets with a single autoregressive neural sequence model trained for each, naturally limiting the breadth of conclusions that we can draw. However, given that the results are consistent and validated by algorithm-independent entropy and efficiency analyses, we believe these findings have general validity and provide a useful starting point for others interested in the problem of querying neural sequence models. Another potential limitation is that our four datasets have relatively small vocabularies ($V = O(10^2)$, small by NLP standards at least); this choice was largely driven by computational limitations in terms of being able to conduct conclusive experiments (e.g. averaging over many event histories). Our (limited) results with GPT-2 provide clues on what may be achievable with much larger vocabularies: systematic analysis of querying for such models is a natural target for future work. Our work also does not address the issue of model inaccuracy: our results are entirely focused on computing query estimates in terms of a model's distribution instead of the data-generating process. Exploring the effect of miscalibration errors in autoregressive models on k-step-ahead query estimates is a promising avenue for future work.

Potential Negative Societal Impacts: Since the focus of this work is making predictions with data that is typically generated by individuals (language, online activity), there is naturally a potential for abuse of such methods. For example, if the underlying model in a system is miscalibrated, decisions could be made that negatively impact individuals. Examples include recommending a student be dropped from an online course if the model incorrectly predicts they will not participate in course modules. Even if the underlying model is well-calibrated, predictive queries could potentially be used in a proactive manner to bias decisions against individuals whose event sequences are atypical, e.g., in a chatbot context, inaccurately predicting future language patterns for certain individuals, leading to interruption and generation of an inappropriate response.

Acknowledgements: We thank the NeurIPS reviewers for their suggestions on improving the original version of this paper. This work was supported by National Science Foundation Graduate Research Fellowship grant DGE-1839285 (AB and SS), by an NSF CAREER Award (SM), by the National Science Foundation under award numbers 1900644 (PS), 2003237 (SM), and 2007719 (SM), by the National Institute of Health under awards R01-AG065330-02S1 and R01-LM013344 (PS), by the Department of Energy under grant DE-SC0022331 (SM), by the HPI Research Center in Machine Learning and Data Science at UC Irvine (SS), and by Qualcomm Faculty awards (PS and SM).

References

Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W. Bruce Croft. Context-aware target apps selection and recommendation for enhancing personal mobile assistants. *ACM Trans. Inf. Syst.*, 39(3):1 – 30, May 2021.

Alan Amin, Eli N Weinstein, and Debora Marks. A generative nonparametric Bayesian model for whole genomes. *Advances in Neural Information Processing Systems*, 34:27798–27812, 2021.

Jeff Bilmes. Dynamic graphical models. *IEEE Signal Processing Magazine*, 27(6):29–42, 2010.

Andy Brown, Aaron Tuor, Brian Hutchinson, and Nicole Nichols. Recurrent neural network attention mechanisms for interpretable system log anomaly detection. In *Proceedings of the First Workshop on Machine Learning for Computing Systems*, pages 1–8, 2018.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Yining Chen, Sorcha Gilroy, Andreas Maletti, Jonathan May, and Kevin Knight. Recurrent neural networks as weighted language recognizers. In *NAACL-HLT*, pages 2261–2271, 01 2018. doi: 10.18653/v1/N18-1205.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2019.

Iris AM Huijben, Wouter Kool, Max Benedikt Paulus, and Ruud JG Van Sloun. A review of the Gumbel-max trick and its extensions for discrete stochasticity in machine learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022, to appear.

John G Kemeny and J Laurie Snell. *Finite Markov Chains*. Springer, 3rd edition, 1983.

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, Cambridge, MA, 2009.

Daphne Koller, Nir Friedman, Lise Getoor, and Ben Taskar. Graphical models in a nutshell. In *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, MA, 2007.

Wouter Kool, Herke Van Hoof, and Max Welling. Stochastic beams and where to find them: The Gumbel-top-k trick for sampling sequences without replacement. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3499–3508. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/kool19a.html`.

Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1269–1278. ACM, 2019.

Changhee Lee, Jinsung Yoon, and Mihaela Van Der Schaar. Dynamic-deephit: A deep learning approach for dynamic survival analysis with competing risks based on longitudinal data. *IEEE Transactions on Biomedical Engineering*, 67(1):122–133, 2019.

Clara Meister and Ryan Cotterell. Language model evaluation beyond perplexity. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5328–5339, 2021.

Clara Isabel Meister, Afra Amini, Tim Vieira, and Ryan Cotterell. Conditional Poisson stochastic beams. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 664–681. Association for Computational Linguistics, 2021.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *Arxiv Preprint*, arXiv:1609.07843, 2017.

Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197, Hong Kong, China, November 2019. Association for Computational Linguistics.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. URL `https://openai.com/blog/better-language-models/`.

Uri Shaham and Omer Levy. What do you get when you cross beam search with nucleus sampling? *arXiv preprint arXiv:2107.09729*, 2022.

William Shakespeare. The Complete Works of William Shakespeare: Project Gutenberg. `https://ocw.mit.edu/ans7870/6/6.006/s08/lecturenotes/files/t8.shakespeare.txt`. Accessed: 2020-04-11.

Kensen Shi, David Bieber, and Charles Sutton. Incremental sampling without replacement for sequence models. In *International Conference on Machine Learning*, pages 8785–8795. PMLR, 2020.

Eui Chul Shin, Miltiadis Allamanis, Marc Brockschmidt, and Oleksandr Polozov. Program synthesis and semantic parsing with learned code idioms. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 10825–10835, Red Hook, NY, USA, 2019. Curran Associates Inc.

Jung-Eun Shin, Adam J Riesselman, Aaron W Kollasch, Conor McMahon, Elana Simon, Chris Sander, Aashish Manglik, Andrew C Kruse, and Debora S Marks. Protein design and variant prediction using autoregressive generative models. *Nature Communications*, 12(1):1–11, 2021.

Øivind Skare, Erik Bølviken, and Lars Holden. Improved sampling-importance resampling and reduced bias importance sampling. *Scandinavian Journal of Statistics*, 30(4):719–737, 2003.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 27:3104 – 3112, 2014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30:6000–6010, 2017.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, 2020. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2020.emnlp-demos.6`.

Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. Recurrent recommender networks. In *Proceedings of the tenth ACM international conference on web search and data mining*, pages 495–503, 2017.

Yonghui Wu, Mike Schuster, Z. Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason R. Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiv Preprint arXiv:1609.08144*, abs/1609.08144, 2016.

Jiacheng Xu and Greg Durrett. Massive-scale decoding for text generation using lattices. *arXiv preprint arXiv:2112.07660*, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 6.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 6.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] We mention computational resources, PII considerations, and other ethical factors in our paper directly.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 3, 4
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Included in supplement as a .zip file
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] We gather data for multiple query histories and report median results for a given experimental setting.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 5.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We cite all public datasets utilized.
   (b) Did you mention the license of the assets? [Yes] Datasets used are all publicly available and the specific license is noted in the Appendix for each.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] Datasets are publicly available.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] See Section 5.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Query Estimation Process and Examples (Section 1 in paper)

Below, we outline in clear terms the process of conducting query estimation experiments that we leverage in the main paper. Furthermore, we include demonstrative examples of potential queries of interest.

Query Estimation Process

1. Sample a sequence x of length n from the test set.
2. Using a model p_θ that was trained on the training split of the dataset, condition on the first 5 elements, $x_{1:5}$.
3. Then, using the proposed method of interest (e.g., importance sampling, coverage-based beam search, etc.), approximate the query of interest for the future continuation of the sequence–typically K steps into the future, $x_{6:6+K}$. As mentioned in lines 256 and 257 of the main paper, the main experiments pertain to hitting time queries $\tau(a) = K$ where $K = 3, \ldots, 11$ and a is determined as being equal to the actual value of x_{6+K} from the sampled sequence. We ensure $6 + K \leq n$.
4. Repeat steps 1-3 over 1000 sequences (unless the dataset is WikiText, in which case do 100 sequences).
5. Against either p determined by absolute ground truth, or pseudo-ground truth p computed via sampling, compute relative absolute error (RAE) of the form $\frac{|\hat{p}-p|}{p}$. For a given dataset, query estimation method, and budget, compute the median RAE over all 1000 (or 100 in the case of WikiTest) sampled sequences, where a sampled sequence is a history-target tuple $[\mathcal{H}, a]$.

MOOCs Query Example:

- **History** $\mathcal{H} = [\texttt{log on}, \texttt{ open assignment}, \texttt{watch lecture}]$
- **Query Class**: "A" before "B" query (Q4)
- **Formalism**: $p^*_\theta(\tau(A) < \tau(B)) = p^*_\theta(\tau(\texttt{purchase}) < \tau(\texttt{log off}))$
- **Description**: Given a user's online class engagement history, what is the probability that they turn in their assignment before they navigate away from the page?

Shakespeare Query Example:

- **History** $\mathcal{H} =$ ["t", "h", "o", "u", "<space>"]
- **Query Class**: Combination query (Q5)
- **Formalism**: $p^*_\theta(N_{\texttt{vowel}}(10) = 4)$
- **Description**: In the next 10 characters, what is the probability that 4 of them are vowels?

B Query Decomposition (Section 3 in paper)

All of the questions posed in Table 1 in the main paper can be decomposed into readily available components that our model p_θ can estimate. Should we not utilize the proposed query structure ($\mathcal{Q} = \cup_i \mathcal{V}_1^{(i)} \times \cdots \times \mathcal{V}_K^{(i)}$) then these are the derivations that would need to be performed individually in order to compute these values exactly on a case-by-case basis.

$\boldsymbol{\mathcal{Q}_1}$ $\mathbb{P}^*(X_1)$ is already naturally in a form that our model can directly estimate due to the autoregressive factorization imposed by the architecture: $p^*_\theta(X_1)$. Furthermore, if we are interested in any potential continuing sequence $X_{1:K} = x_{1:K}$ this can easily be computed via $p^*_\theta(X_{1:K} = x_{1:K}) = \prod_{k=1}^{K} p^*_\theta(X_k = x_k | X_{<k} = x_{<k})$.

$\mathcal{Q}_2$ To evaluate $p^*_\theta(X_K)$, terms $X_{<K}$ need to be marginalized out. This is naturally represented like so:

$$\begin{aligned} p^*_\theta(X_K) &= \mathbb{E}_{x_{<K} \sim p^*_\theta(X_{<K})} \left[p^*_\theta(X_K | X_{<K} = x_{<K}) \right] \\ &= \sum_{x_{<K} \in \mathbb{V}^{K-1}} p^*_\theta(X_K, X_{<K} = x_{<K}) \end{aligned}$$

It is helpful to show both the exact summation form as well as the expected value representation as both will be useful in Section 4.

$\mathcal{Q}_3$ The "hitting time" or the next occurrence of a specific event type $a \in \mathbb{V}$ is defined as $\tau(a)$. Evaluating the distribution of the hitting time with our model can be done like so:

$$\begin{aligned} p^*_\theta(\tau(a) = K) &= p^*_\theta(X_K = a, X_{<K} \neq a), \\ &= \sum_{x_{<K} \in \{\mathbb{V} \setminus \{a\}\}^{K-1}} p^*_\theta(X_K = a, X_{<K} = x_{<K}). \end{aligned}$$

The value $a \in \mathbb{V}$ can be easily replaced with a set of values $A \subset \mathbb{V}$ in these representations.

$$\begin{aligned} p^*_\theta(\tau(A) = k) &= p^*_\theta(X_K \in A, X_{<K} \in (\mathbb{V} \setminus A)^{K-1}), \\ &= \sum_{x_{<K} \in \{\mathbb{V} \setminus A\}^{K-1}} \sum_{a \in A} p^*_\theta(X_K = a, X_{<K} = x_{<K}) \end{aligned}$$

Interestingly, we can see that Q3 is a generalization of Q2 by noting that they are identical when $A = \{\}$.

$\mathcal{Q}_4$ To evaluate $p^*_\theta(\tau(a) < \tau(b))$, we must consider the possible instances where this condition is fulfilled. In doing so, we end up evaluating multiple queries similar in form to Q3, like so:

$$\begin{aligned} p^*_\theta(\tau(a) < \tau(b)) &= \sum_{k=1}^{\infty} p^*_\theta(\tau(a) = k, \tau(b) > k) \\ &= \sum_{k=1}^{\infty} p^*_\theta(X_k = a, X_{<k} \in (\mathbb{V} \setminus \{a, b\})^{k-1}) \end{aligned} \tag{5}$$

In practice, computing this exactly is intractable due to it being an infinite sum. There are two potential approaches one could take to subvert this. The first of which is to ask a slightly different query:

$$\begin{aligned} p^*_\theta(\tau(a) < \tau(b) | \tau(a) \leq K) &= \frac{p^*_\theta(\tau(a) < \tau(b), \tau(a) \leq K)}{p^*_\theta(\tau(a) \leq K, \mathcal{H}_T)} \\ &= \frac{\sum_{k=1}^{K} p^*_\theta(X_k = a, X_{<k} \in (\mathbb{V} \setminus \{a, b\})^{k-1})}{\sum_{k=1}^{K} p^*_\theta(\tau(a) = k)}. \end{aligned}$$

The other option is to produce a lower bound on this expression by evaluating the sum in Eq. (5) for the first K terms. We can achieve error bounds on this estimate by noting that $p^*_\theta(\tau(a) < \tau(b)) + p^*_\theta(\tau(a) > \tau(b)) = 1$. As such, if we evaluate Eq. (5) up to K terms for both $p^*_\theta(\tau(a) < \tau(b))$ and $p^*_\theta(\tau(b) < \tau(a))$, the difference between the sums will be the maximum error either lower bound can have.

Similar to Q3, we can also ask this query with sets $A \perp B \subset \mathbb{V}$ instead of values a, b.

$$\begin{aligned} p^*_\theta(\tau(A) < \tau(B)) &= \sum_{k=1}^{\infty} p^*_\theta(\tau(A) = k, \tau(B) > k) \\ &= \sum_{k=1}^{\infty} \sum_{a \in A} p^*_\theta(X_k = a, X_{<k} \in (\mathbb{V} \setminus (A \cup B))^{k-1}) \end{aligned}$$

$\mathcal{Q}_5$ Evaluating $p^*_\theta(N_a(K) = n)$ also involves decomposing this value of interest into statements involving hitting times.

Let $\tau^{(l)}(a)$ be the lth hitting time of a specific event of interest a. In other words, the time of the lth occurrence of a. Assuming $n \leq K$:

$$
\begin{aligned}
&p^*_\theta(N_a(K) = n) \\
&\quad = \sum_{i_1 < i_2 < \cdots < i_n \leq K} p^*_\theta(\tau^{(1)}(a) = i_1, \tau^{(2)}(a) = i_2, \ldots, \tau^{(n)}(a) = i_n, \tau^{(n+1)}(a) > K) \\
&\quad = \sum_{i_1 < i_2 < \cdots < i_n \leq K} p^*_\theta(X_{i_1} = \tau^{(1)}(a) = i_1, \tau^{(2)}(a) = i_2, \ldots, \tau^{(n)}(a) = i_n, \tau^{(n+1)}(a) > K)
\end{aligned}
$$

Approximation Errors With queries that fall under the Q5 category (e.g., how likely will the event a happen n times in the next K steps? $p^*_\theta(N_a(K) = n)$), while in the limiting case C(K,n) will become the dominant scaling factor for the size of the query space, in many circumstances this actually isn't as impactful as one might expect. For instance, it can be shown that the query space of a Q5 query is larger than that of a hitting time query (Q3) evaluated one step further (at $K + 1$) if $\sqrt[n]{C(K, n)} > V$ where V is the vocabulary size. Looking at our main set of experiments from Section 5, we can see that when evaluating hitting time queries up to 11 steps into the future ($K = 11$), these queries had a slightly larger query space of than that of $p^*_\theta(N_a(K) = n)$ for $K = 10, \forall_{0 \leq n \leq K}$ and thus can be seen as comparable in complexity. This behavior easily extends to larger values of K (e.g., $K \approx 100$), especially for larger vocabulary sizes. This is due to the nth root dramatically slowing down the factorial growth in the inequality shared above.

C Complexity of Predictive Querying with First-Order Markov Models (Section 3 in paper)

As a point of reference for neural sequence models we summarize below the cost of various queries for Markov models (using basic results from the theory of finite Markov chains, e.g., see [Kemeny and Snell, 1983]). Consider a first-order ergodic homogeneous Markov chain with a $V \times V$ transition matrix with elements $p(x_{k+1}|x_k)$. We analyze below the complexity of various queries for such a chain, conditioned on the current observation $X_0 = v$, where $v \in \mathbb{V}$ (this is analogous to conditioning on $\mathcal{H}$ for neural sequence models). We use $p(\cdot|v)$ as shorthand for $p(\cdot|X_0 = v)$.

Q2: computing $p(X_k|v)$ This can be computed recursively by computing the conditional distrubution $p(X_1|v)$, then $p(X_2|v) = \sum_{v'} p(X_2|v')p(v'|v)$, and so on, resulting in k matrix multiplications, with complexity $O(kV^2)$.

Q3: computing $p(\tau(a) = k|v)$ For finite k, we can compute the result for all values $k' \leq k$ using $k - 1$ matrix multiplications where, at each step $k' = 2, \ldots, k - 1$, all states (events) except a are marginalized over. Thus, the complexity is $O(kV^2)$.

In general it is straightforward to show that

$$p(\tau(a) = k|v) = p(a|v), \quad k = 1$$

and

$$p(\tau(a) = k|v) = p(a|\bar{a})p(\bar{a}|v)p(\bar{a}|\bar{a})^{(k-2)}, \quad k > 1$$

where $p(a|\bar{a})$ is the probability of transitioning to state a given that the chain is not in state a. Computation of $p(a|\bar{a})$ requires knowledge of the steady-state probabilities of the chain π_a and $\pi_{\bar{a}} = \sum_{v' \neq a} \pi_{v'}$. Computing the steady-state probabilities requires inversion of a $V \times \mathbb{V}$ matrix, with a complexity of $O(V^3)$. This general solution will be faster to compute than the version using matrix multiplication (up to horizon k) whenever $k > V$.

Q4: computing $p(\tau(a) < \tau(b)|v)$ Let c be the set of all states (events) in $\mathbb{V}$ except for a and b. It is straightforward to show that

$$p(\tau(a) < \tau(b)|v) = p(a|v) + p(c|v)\frac{p(a|c)}{p(a|c) + p(b|c)}$$

and

$$p(\tau(b) < \tau(a)|v) = p(b|v) + p(c|v)\frac{p(b|c)}{p(a|c) + p(b|c)}$$

where $p(a|c)$ and $p(b|c)$ are the probabilities of transitioning to a and b, respectively, given that the system is currently in a state that is neither a or b. Computing these probabilities again requires knowledge of the steady-state probabilities π_a, π_b, π_c, resulting in $O(V^3)$ time complexity.

General Queries and Higher Order Markov Models For simplicity, we will assume that a query takes the form $\mathcal{Q} = \mathcal{V}_1 \times \cdots \times \mathcal{V}_K$ and we are interested in $p(X_{1:K} \in \mathcal{Q}|\mathcal{H})$ for a mth-order Markov model p. This model can be defined with an $(m+1)$-dimensional tensor $\Pi \in \mathbb{R}^{V \times \cdots \times V}$ with elements $\pi_{i_1,\ldots,i_m,i_{m+1}}$ such that $\sum_{j=1}^{V} \pi_{i_1,\ldots,i_m,j} = 1$ for all $i_1, \ldots, i_m \in \mathbb{V}$. Alternatively, $\pi_{i_1,\ldots,i_m,i_{m+1}} = p(X_{j+m+1} = i_{m+1}|X_{j+1:j+m} = i_{1:m})$ for $j \geq 0$.

To marginalize out X_{m+1} and compute the conditional distribution $p(X_{m+2}|X_{1:m})$, that requires the following computation:

$$\begin{aligned}
p(X_{m+2} = x_{m+2}|X_{1:m} = x_{1:m}) &= \sum_{v \in \mathbb{V}} p(X_{m+2} = a, X_{m+1} = v|X_{1:m} = x_{1:m}) \\
&= \sum_{v \in \mathbb{V}} p(X_{m+2} = x_{m+2}|X_{m+1} = v, X_{2:m} = x_{2:m}) p(X_{m+1} = v|X_{1:m} = x_{1:m}) \\
&= \sum_{v \in \mathbb{V}} \pi_{x_2,\ldots,x_m,v,a} \pi_{x_1,\ldots,x_{m-1},x_m,v}
\end{aligned}$$

If we perform this over all values of $x_{1:m}$, we can construct a new transition tensor representing $p(X_{m+2}|X_{1:m})$. We will denote this new tensor as being equal to $\Pi \otimes \Pi$ where $(\Pi \otimes \Pi)_{i_1,\ldots,i_m,v} = p(X_{j+m+2} = v|X_{j+1:j+m} = i_{1:m})$. This operation has a computation complexity of $\mathcal{O}(V^{m+1})$.

This special product can be done repeatedly to further marginalize out. For instance, performing this operation on Π $(n-1)$-times results in $(\Pi \otimes \cdots \otimes \Pi)_{i_1,\ldots,i_m,v} = p(X_{j+m+n} = v|X_{j+1:j+m} = i_{1:m})$, thus marginalizing out $n-1$ terms: $X_{j+m+1:j+m+n-1}$.

Transitioning into a restricted space $\mathcal{V}$ can be done easily by defining a restricted transition tensor $\Pi_{\mathcal{V}}$ such that $(\Pi_{\mathcal{V}})_{i_1,\ldots,i_m,v} \propto \pi_{i_1,\ldots,i_m,v}$ if $v \in \mathbb{V}$, otherwise $(\Pi_{\mathcal{V}})_{i_1,\ldots,i_m,v} = 0$ for all $i_1, \ldots, i_m, v \in \mathbb{V}$.

With this, we have everything we need to compute $p(X_{1:K} \in \mathcal{Q}|\mathcal{H})$. If the last m-values of the history $\mathcal{H}$ are equal to $i_1, \ldots, i_m$, then:

$$\begin{aligned}
p(X_{1:K} \in \mathcal{Q}|\mathcal{H}) &= \prod_{k=1}^{K} p(X_k \in \mathcal{V}_k|X_{<k} \in \mathcal{V}_1 \times \cdots \times \mathcal{V}_{k-1}) \\
&= \prod_{k=1}^{K} \sum_{v \in \mathcal{V}_k} p(X_k = v|X_{<k} \in \mathcal{V}_1 \times \cdots \times \mathcal{V}_{k-1}) \\
&= \prod_{k=1}^{K} \sum_{v \in \mathcal{V}_k} (\Pi_{\mathcal{V}_1} \otimes \cdots \otimes \Pi_{\mathcal{V}_k})_{i_1,\ldots,i_m,v}.
\end{aligned}$$

The dominant factor in the computational complexity of this is computing all of the special products of Π. As such, this has a total computational complexity of $\mathcal{O}((k-1)V^{m+1})$.

D Details for the Hybrid Method (Section 4 in paper)

The hybrid method estimates an arbitrary probabilistic query $p^*_\theta(X_{1:K} \in \mathcal{Q})$ by first using some variant of beam search (in our results we use our proposed *tail-splitting* beam search, see Section 3) to find a lower bound $p^*_\theta(X_{1:K} \in \mathcal{B})$, and then using a sampling method (we use importance sampling, see Section 3) to estimate the remainder $p^*_\theta(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B})$. The only complicated portion of this process lies in how we derive an acceptable proposal distribution for the sampling phase that respects the domain $\mathcal{Q} \setminus \mathcal{B}$. The following paragraphs detail how we construct such a distribution.

Viewing $\mathbb{V}^K$ and p^*_θ as Trees In the space of all possible sequences of length K, $X_{1:K} \in \mathbb{V}^K$, one can represent these sequences as paths in a tree. Each node in this tree represents a single element in a sequence $X_k \in \mathbb{V}$ with depth k, with parent and children nodes representing previous and potential future values in the sequence respectively. The root node either represents the very beginning of a sequence, or a concrete history $\mathcal{H}$ to condition on.

This tree can be augmented into a probabilistic one by defining edges between nodes as the conditional probability of a child node being next in a sequence, conditioned on all ancestors of that child. These probabilities are naturally determined by $p^*_\theta(X_k = x_k | X_{<k} = x_{<k})$ where x_k is the child node value and $x_{<k}$ are the ancestors' values.

Building the Tree Any subset of $\mathbb{V}^K$ can also be represented as a tree, and in fact will be a sub-tree of the one that represents $\mathbb{V}^K$. As such, there exists a tree that represents $\mathcal{Q}$. Our usual proposal distribution q is a natural source of conditional probabilities for the edges. While none of these trees with their edge weights are fully known ahead of time, we do explore and uncover them through the process of beam search. As such, during the beam search phase of the hybrid method we keep track of any conditional distributions $q(X_k = x_k | X_{<k} = x_{<k})$ that are computed and use them to construct a partial view of the tree for $\mathcal{Q}$. Note that the end result of this process is a tree that will likely have many paths that do not fully reach depth K; however, there will be at least $|\mathcal{B}|$ many that do.

For our purposes, it is also useful to keep track of $p^*_\theta(X_k = x_k | X < k = x_{<k})$ over this restricted set, as well as model byproducts such as hidden states to reduce computation redundancy later.

Pruning the Tree After beam search has completed, we are left with a resulting set of beams $\mathcal{B}$ and a probabilistic tree representing q. We would now like to alter this tree such that its weights represent $q_{\mathcal{B}}(X_{1:K} = x_{1:K}) := q(X_{1:K} = x_{1:K} | X_{1:K} \notin \mathcal{B})$. This alteration can be accomplished by adjusting the edge weights in the tree recursively as detailed below. New weight assignments will be denoted by q' to differentiate from old weights q. The steps to the procedure are defined as follows:

1. At the final depth K, assign edge weights $q'(X_K = x_K | X_{<K} = x_{<K}) = 0$ for all $x_{1:K} \in \mathcal{B}$. All other edge weights in the final depth will have new weights $q'(X_K = x_K | X_{<K} = x_{<K}) = q(X_K = x_K | X_{<K} = x_{<K})$.
2. At the next layer above with depth $k = K - 1$, assign edge weights as $q'(X_k = x_k | X_{<k} = x_{<k}) = q(X_k = x_k | X_{<k} = x_{<k}) \sum_{v \in \mathbb{V}} q'(X_{k+1} = v | X_{\leq k} = x_{\leq k})$ for all sub-sequences $x_{1:k} \in \mathcal{B}$.
3. Repeat step 2 iteratively for $k = K - 2, K - 3, \ldots, 2, 1$.
4. Finally, normalize every conditional distribution for every node whose children edges were altered such that they each sum to 1.

After these steps are completed, $q_{\mathcal{B}}(X_{1:K} = x_{1:K}) = \prod_{k=1}^{K} q'(X_k = x_k | X_{<k} = x_{<k})$. Note that weights related to sequences that were not discovered during beam search, and are thus not in the tree, are not altered and still match the original proposal distribution q. As such, to sample sequences from the tree, we start at the root node and sample from each successive conditional distribution until either depth K or a leaf node at depth $k < K$ is reached. In the former scenario, the sampling is complete. In the latter, the remaining values of the sequence are sampled from $q(\cdot | X_{\leq k} = x_{\leq k})$ like usual.

E Variance of Estimates from the Hybrid Method (Section 4 in paper)

We are assuming to be under the hybrid method regime where a collection of sequences $\mathcal{B} \subset \mathcal{Q}$ relevant to answering $p^*_\theta(X_{1:K} \in \mathcal{Q})$ have been deterministically found and are interested in using sampling methods to estimate the remainder $p^*_\theta(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B})$. For brevity, we will assume that $\mathcal{Q} = \mathcal{V}_1 \times \cdots \times \mathcal{V}_K$. As mentioned in the previous section, we leverage our originally presented proposal distribution q by further restricting the domain to $\mathcal{Q} \setminus \mathcal{B}$ in order to be used in this scenario. This will be represented by

$$\begin{aligned} q_{\mathcal{B}}(X_{1:K} = x_{1:K}) &:= q(X_{1:K} = x_{1:K} | X_{1:K} \notin \mathcal{B}) \\ &= \frac{q(X_{1:K} = x_{1:K}) \mathbb{1}(x_{1:K} \in \mathcal{Q} \setminus \mathcal{B})}{1 - q(X_{1:K} \in \mathcal{B})}. \end{aligned}$$

Note that an associated autoregressive form $q_{\mathcal{B}}(X_k = x_k | X_{<k} = x_{<k})$ exists and is well defined; however, the exact definition is a bit unwieldy. Please refer to the previous section for more details.

Using this proposal distribution, we can easily estimate the remaining probability:

$$\begin{aligned} p_\theta^*(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B}) &= \mathbb{E}_{x_{1:K} \sim q_{\mathcal{B}}} \left[\frac{p_\theta^*(X_{1:K} = x_{1:K})}{q_{\mathcal{B}}(X_{1:K} = x_{1:K})} \right] \\ &\approx \frac{1}{M} \sum_{m=1}^{M} \frac{p_\theta^*(X_{1:K} = x_{1:K}^{(m)})}{q_{\mathcal{B}}(X_{1:K} = x_{1:K}^{(m)})} \qquad \text{for } x_{1:K}^{(1)}, \ldots, x_{1:K}^{(M)} \overset{iid}{\sim} q_{\mathcal{B}}. \end{aligned}$$

Let $\omega_{\mathcal{B}}(x_{1:K}) := \frac{p_\theta^*(X_{1:K}=x_{1:K})}{q_{\mathcal{B}}(X_{1:K}=x_{1:K})}$ and for brevity we will refer to $p_\theta^*(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B})$ as $\bar{\omega}_{\mathcal{B}}$.

If we assume $\mathcal{B}' = \mathcal{B} \cup \{\hat{x}_{1:K}\}$ for some $\hat{x}_{1:K} \in \mathcal{Q} \setminus \mathcal{B}$, then it is interesting to determine when exactly there will be a reduction in sampling variance for $p_\theta^*(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B}')$ versus $p_\theta^*(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B})$ as this will give insight into when the hybrid method is successful. In other words, we would like to show when the following inequality holds true:

$$\Delta_{\text{Var}} := \text{Var}_{x_{1:K} \sim q_{\mathcal{B}}} \left[\omega_{\mathcal{B}}(x_{1:K}) \right] - \text{Var}_{x_{1:K} \sim q_{\mathcal{B}'}} \left[\omega_{\mathcal{B}'}(x_{1:K}) \right] \geq 0$$

If this is true for a given $\hat{x}_{1:K}$, then this finding can be applied recursively for more general (but still possibly restricted) $\mathcal{B}' \supset \mathcal{B}$.

$$\begin{aligned} \text{Var}_{x_{1:K} \sim q_{\mathcal{B}}} \left[\omega_{\mathcal{B}}(x_{1:K}) \right] &= \mathbb{E}_{x_{1:K} \sim q_{\mathcal{B}}} \left[\left(\omega_{\mathcal{B}}(x_{1:K}) - \bar{\omega}_{\mathcal{B}} \right)^2 \right] \\ &= \mathbb{E}_{x_{1:K} \sim q_{\mathcal{B}}} \left[\omega_{\mathcal{B}}(x_{1:K})^2 \right] - \bar{\omega}_{\mathcal{B}}^2 \\ &= \sum_{x_{1:K} \in \mathcal{Q} \setminus \mathcal{B}} q_{\mathcal{B}}(x_{1:K}) \omega_{\mathcal{B}}(x_{1:K})^2 - \bar{\omega}_{\mathcal{B}}^2 \end{aligned}$$

It then follows that

$$\begin{aligned} \Delta_{\text{Var}} &= \bar{\omega}_{\mathcal{B}'}^2 - \bar{\omega}_{\mathcal{B}}^2 + \sum_{x_{1:K} \in \mathcal{Q} \setminus \mathcal{B}} q_{\mathcal{B}}(x_{1:K}) \omega_{\mathcal{B}}(x_{1:K})^2 - \sum_{x_{1:K} \in \mathcal{Q} \setminus \mathcal{B}'} q_{\mathcal{B}'}(x_{1:K}) \omega_{\mathcal{B}'}(x_{1:K})^2 \\ &= (\bar{\omega}_{\mathcal{B}'}^2 - \bar{\omega}_{\mathcal{B}}^2) + q_{\mathcal{B}}(\hat{x}_{1:K}) \omega_{\mathcal{B}}(\hat{x}_{1:K})^2 + \\ &\qquad \sum_{x_{1:K} \in \mathcal{Q} \setminus \mathcal{B}'} \left[q_{\mathcal{B}}(x_{1:K}) \omega_{\mathcal{B}}(x_{1:K})^2 - q_{\mathcal{B}'}(x_{1:K}) \omega_{\mathcal{B}'}(x_{1:K})^2 \right] \end{aligned} \tag{6}$$

We will now analyze each of the three terms in Eq. (6) to determine when $\Delta_{\text{Var}} \geq 0$. For the first term, it follows that:

$$\begin{aligned} \bar{\omega}_{\mathcal{B}'} + \bar{\omega}_{\mathcal{B}} &= p_\theta^*(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B}') + p_\theta^*(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B}) \\ &= 2p_\theta^*(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B}') + p_\theta^*(\hat{x}_{1:K}) \\ \bar{\omega}_{\mathcal{B}'} - \bar{\omega}_{\mathcal{B}} &= p_\theta^*(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B}') - p_\theta^*(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B}) \\ &= -p_\theta^*(\hat{x}_{1:K}) \\ \implies \bar{\omega}_{\mathcal{B}'}^2 - \bar{\omega}_{\mathcal{B}}^2 &= -p_\theta^*(\hat{x}_{1:K}) \left(2p_\theta^*(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B}') - p_\theta^*(\hat{x}_{1:K}) \right) \end{aligned} \tag{7}$$

The other two terms in Eq. (6) must sum to a positive value with as large or larger magnitude to Eq. (7) for $\Delta_{\text{Var}} \geq 0$. Looking at the second term, we see that

$$\begin{aligned} q_{\mathcal{B}}(\hat{x}_{1:K}) \omega_{\mathcal{B}}(\hat{x}_{1:K})^2 &= \frac{p_\theta^*(\hat{x}_{1:K})^2}{q_{\mathcal{B}}(\hat{x}_{1:K})} \\ &= p_\theta^*(\hat{x}_{1:K})(1 - q(X_{1:K} \in \mathcal{B})) \prod_{k=1}^{K} p_\theta^*(X_k \in \mathcal{V}_k | X_{<k} = \hat{x}_{<k}) \geq 0. \end{aligned}$$

This inequality becomes strict should $p_\theta^*(\hat{x}_{1:K}) > 0$. We will now look at the final summation in Eq. (6):

$$
\begin{aligned}
&\sum_{x_{1:K} \in \mathcal{Q} \setminus \mathcal{B}'} \left[q_{\mathcal{B}}(x_{1:K}) \omega_{\mathcal{B}}(x_{1:K})^2 - q_{\mathcal{B}'}(x_{1:K}) \omega_{\mathcal{B}'}(x_{1:K})^2 \right] \\
&= \sum_{x_{1:K} \in \mathcal{Q} \setminus \mathcal{B}'} \left(\frac{p_\theta^*(x_{1:K})^2}{q_{\mathcal{B}}(x_{1:K})} - \frac{p_\theta^*(x_{1:K})^2}{q_{\mathcal{B}'}(x_{1:K})} \right) \\
&= \sum_{x_{1:K} \in \mathcal{Q} \setminus \mathcal{B}'} p_\theta^*(x_{1:K}) \left(q(X_{1:K} \in \mathcal{B}') - q(X_{1:K} \in \mathcal{B}) \right) \prod_{k=1}^{K} p_\theta^*(X_k \in \mathcal{V}_k | X_{<k} = x_{<k}) \\
&= q(\hat{x}_{1:K}) \sum_{x_{1:K} \in \mathcal{Q} \setminus \mathcal{B}'} p_\theta^*(x_{1:K}) \prod_{k=1}^{K} p_\theta^*(X_k \in \mathcal{V}_k | X_{<k} = x_{<k}) \\
&= \frac{p_\theta^*(\hat{x}_{1:K})}{\prod_{k=1}^{K} p_\theta^*(X_k \in \mathcal{V}_k | X_{<k} = \hat{x}_{<k})} \sum_{x_{1:K} \in \mathcal{Q} \setminus \mathcal{B}'} p_\theta^*(x_{1:K}) \prod_{k=1}^{K} p_\theta^*(X_k \in \mathcal{V}_k | X_{<k} = x_{<k}).
\end{aligned}
$$

Since all terms in Eq. (6) have a common factor $p_\theta^*(\hat{x}_{1:K})$, we can see that $\Delta_{\text{Var}} \geq 0$ iff the following holds:

$$
\begin{aligned}
2p_\theta^*(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B}') - \frac{1}{\rho(\hat{x}_{1:K})} \sum_{x_{1:K} \in \mathcal{Q} \setminus \mathcal{B}'} p_\theta^*(x_{1:K}) \rho(x_{1:K}) \\
\leq (1 - q(X_{1:K} \in \mathcal{B})) \rho(\hat{x}_{1:K}) + p_\theta^*(\hat{x}_{1:K})
\end{aligned} \tag{8}
$$

for $\rho(x_{1:K}) = \prod_{k=1}^{K} p_\theta^*(X_k \in \mathcal{V}_k | X_{<k} = x_{<k})$. Should this hold true, then by taking $\mathcal{B}'$ instead of $\mathcal{B}$ during the beam search segment of the hybrid approach would the variance of the sampling subroutine reduce. Generalizing this further, it is not guaranteed that the hybrid estimate will have a lower variance than regular importance sampling; however, our experimental results across a variety of settings (see Fig. 3 in the main paper) seem to indicate that the variance is reduced on average.

All of the terms to the left of the inequality in Eq. (8) are quantities that would require either expansive computations or estimation in order to know their values. Conversely, all the values to the right of the inequality are readily available as a byproduct of beam search. Incorporating this into decision making for our hybrid method is left for future work.

F Proof for Coverage-Based Beam Search Error Upper Bound (Section 4.2 in paper)

Theorem 1. *For a given set of decoded sequences $\mathcal{B} \subset \mathcal{Q}$ with coverage $q(X_{1:K} \in \mathcal{B})$, the error between the true probabilistic query value $p_\theta^*(X_{1:K} \in \mathcal{Q})$ and the lower bound $p_\theta^*(X_{1:K} \in \mathcal{B})$ is bounded above by the complement of the coverage: $1 - q(X_{1:K} \in \mathcal{B})$.*

Proof. For brevity, we will assume that $\mathcal{Q} = \mathcal{V}_1 \times \cdots \times \mathcal{V}_K$ (although this can easily be extended to the general case). Since $q(X_k = x_k | X_{<k} = x_{<k}) = \frac{p_\theta^*(X_k = x_k | X_{<k} = x_{<k}) \mathbb{1}(x_k \in \mathcal{V}_k)}{p_\theta^*(X_k \in \mathcal{V}_k | X_{<k} = x_{<k})}$ and $p_\theta^*(\cdot) \leq 1$, it follows that $q(X_k = x_k | X_{<k} = x_{<k}) \geq p_\theta^*(X_k = x_k | X_{<k} = x_{<k})$. This becomes a strict inequality should $p_\theta^*(X_k \in \mathcal{V}_k | X_{<k} = x_{<k}) < 1$ (which is often the case should $\mathcal{V}_k \subset \mathbb{V}$). Since this holds for arbitrary k and x_k, it then follows that $q(X_{1:K} = x_{1:K}) \geq p_\theta^*(X_{1:K} = x_{1:K})$ for any $x_{1:K} \in \mathcal{Q}$.[6] This inequality becomes strict should any sequence not in the query set have non-zero probability, i.e., $p_\theta^*(X_{1:K} = x_{1:K}) > 0$ for any $x_{1:K} \in \mathbb{V}^K \setminus \mathcal{Q}$.

The target value that we are estimating can be broken up into the following terms:

$$
p_\theta^*(X_{1:K} \in \mathcal{Q}) = p_\theta^*(X_{1:K} \in \mathcal{B}) + p_\theta^*(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B}).
$$

[6] This can be seen by comparing their autoregressive factorizations term by term.

Rearranging these terms yields us the error of our lower bound. We can easily derive an upper bound on the error for an arbitrary set $\mathcal{B}$:

$$\begin{aligned}
p^*_\theta(X_{1:K} \in \mathcal{Q}) - p^*_\theta(X_{1:K} \in \mathcal{B}) &= p^*_\theta(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B}) \\
&= \sum_{x_{1:K} \in \mathcal{Q} \setminus \mathcal{B}} p^*_\theta(X_{1:K} = x_{1:K}) \\
&\leq \sum_{x_{1:K} \in \mathcal{Q} \setminus \mathcal{B}} q(X_{1:K} = x_{1:K}) \\
&= q(X_{1:K} \in \mathcal{Q} \setminus \mathcal{B}) \\
&= q(X_{1:K} \in \mathcal{Q}) - q(X_{1:K} \in \mathcal{B}) \\
&= 1 - q(X_{1:K} \in \mathcal{B}) \leq 1 - \alpha
\end{aligned}$$

where α is the targeted coverage probability used to find $\mathcal{B}$ with coverage-based beam search.[7] This inequality becomes strict should any sequence not in the query set have non-zero probability, i.e., $p^*_\theta(X_{1:K} = x_{1:K}) > 0$ for any $x_{1:K} \in \mathbb{V}^K \setminus \mathcal{Q}$. □

G Experimental Setup and Preparation (Section 5 in paper)

In this section we disclose all dataset preparation and modeling details necessary for reproducing our experimental results.

G.1 Datasets

We evaluate our query estimation methods on three user behavior and two language datasets. We provide details on the preparation and utilization of each below. For all datasets, users are associated with anonymous aliases to remove personally identifiable information (PII).

Reviews [Ni et al., 2019] contains sequences of 233 million timestamped Amazon product reviews spanning from May 1996 to October 2018, with each product belonging to one of 30 product categories. We restrict our consideration of this dataset to reviews generated by users with at least 15 product reviews and products with a defined category, retaining 63 million reviews on which the model was trained. This dataset is publicly available under the Amazon.com Conditions of Use License.

Mobile Apps [Aliannejadi et al., 2021] consists of 3.6 million app usage records from 200 Android users from September 2017 to May 2018, where each event is an interaction of an individual with an application. User behavior spans 87 unique applications, which we use as the vocabulary for events for our experiments. This dataset is released under the Creative Commons License, and all users contain at least 15 mobile app interactions so no data was removed before training.

MOOCs [Kumar et al., 2019] is a dataset of sequences of anonymized user interactions with online course materials from a set of massive open online courses (MOOCs). In total, the dataset includes 97 unique types of interactions. Data from users with fewer than 15 interaction events are not considered, resulting in a dataset of 72% of users and 93% of the events (350,000 interactions) of the original dataset. The MOOCs dataset is available under the MIT License.

Shakespeare We also examine character-level language models, using the complete works of William Shakespeare [Shakespeare], comprising 125,000 lines of text and 67 unique characters and released under the Project Gutenberg License.

WikiText The WikiText-v2 dataset [Merity et al., 2017] includes word-level language data from "verified Good" and featured articles of Wikipedia. All sentences are provided in English and the dataset is available under the Creative Commons Attribution-ShareAlike License.

[7] Note that $q(X_{1:K} \in \mathcal{Q}) = 1$ because by design $q(X_{1:K} = x_{1:K}) = 0$ for every $x_{1:K} \notin \mathcal{Q}$.

G.2 Neural Sequence Models

For all datasets except WikiText (word-level language), we trained a 2-layer LSTM with a dropout rate of 0.3 and the ReLU activation function. Each network was trained against cross entropy loss with the Adaptive Moments (Adam) optimizer initialized with a learning rate of 0.001. A constant learning rate decay schedule and 0.01 warm-up iteration percentage was also used. All LSTM models maintain a hidden state size of 512 except the model for Shakespeare, which possesses a reduced hidden state size of 128 to reflect the size of the dataset. Model checkpoints were collected for all models every 2 epochs, and the checkpoint with the highest validation accuracy was selected to be used for query estimation experiments. All models were trained on NVIDIA GeForce 2080ti GPUs.

For WikiText-v2, we leveraged the GPT-2 medium (350 million parameters) Architecture from HuggingFace with pre-trained weights provided by OpenAI. The WikiText-v2 dataset was preprocessed using the tokenization scheme provided by HuggingFace for GPT-2, assigning numeric token indices to work pieces. No finetuning of GPT-2 is conducted.

G.3 Ground Truth Calculations

Exact computation of ground truth is intractable for queries with large path spaces. We circumvent this issue via the law of large numbers by computing **surrogate ground truth** query estimates with a large computational budget, leveraging the variance of the query's samples as a convergence criterion. Specifically, our algorithm is conducted as follows. We first specify a *minimum* number of samples $S_{\text{low}} = 10000$ to be drawn for the surrogate ground-truth estimate. Once S_{low} samples have been drawn, we compute the variance of our estimate $\hat{p}^*_\theta(X_{1:K} \in \mathcal{Q}) := \frac{1}{S}\sum_{i=1}^{S} \frac{p^*_\theta(X_{1:K}=x^{(i)}_{1:K})}{q(X_{1:K}=x^{(i)}_{1:K})}$ for $x^{(1)}_{1:K}, \ldots, x^{(S)}_{1:K} \overset{iid}{\sim} q(X_{1:K})$:

$$\widehat{\text{Var}}_q\left[\hat{p}^*_\theta(X_{1:K} \in \mathcal{Q})\right] = \frac{1}{S}\sum_{i=1}^{S}\left(\frac{p^*_\theta(X_{1:K} = x^{(i)}_{1:K})}{q(X_{1:K} = x^{(i)}_{1:K})} - \hat{p}^*_\theta(X_{1:K} \in \mathcal{Q})\right)^2$$

We then evaluate $\widehat{\text{Var}}_q\left[\hat{p}^*_\theta(X_{1:K} \in \mathcal{Q})\right]$ every 1000 additional samples until either it drops below tolerance $\delta = 1e^{-7}$ or S meets our maximum sample budget $S_{\text{high}} = 100000$. This procedure is done in all of our experiments in which a method's performance is being compared to a query's ground truth value and exact ground truth cannot be computed due to resource constraints (typically when $K > 4$).

G.4 Model Budget Determination

Each of the estimation methods that have been discussed can use varying amounts of computation depending on their configuration. To ensure an even comparison between methods, we configure them throughout our experiments to have roughly the same amount of computation. We measure amount of computation by the number of **model calls** $f_\theta(h_k)$ used within a given method. This can be controlled directly for all of the methods except the hybrid approach. For this reason, in our experiments we typically use fix the number of samples S for the importance sampling component of the hybrid method, where S is the number of samples used by the hybrid method after conducting tail-splitting beam search. We then use the resulting number of total model calls used by the hybrid method (including both beam search and sampling with S samples) to determine a fixed computation budget (number of model calls to compute $f_\theta(h_k)$) for all other methods. Should the hybrid method not be used in an experiment, the budget is set by determining the number of model calls used in drawing S samples for importance sampling.

H Additional Experimental Details and Results (Section 5 in paper)

This section discusses additional experimental details and results that were not included in section 5 of our main paper due to space constraints. In all experiments, all means and medians reported are with respect to $N_Q = 1000$ randomly selected sequence locations/histories/queries per datapoint in each plot, unless stated otherwise. For each randomly selected current location, the event a used in the query for k steps ahead corresponds to the actual observed event a for k steps ahead.

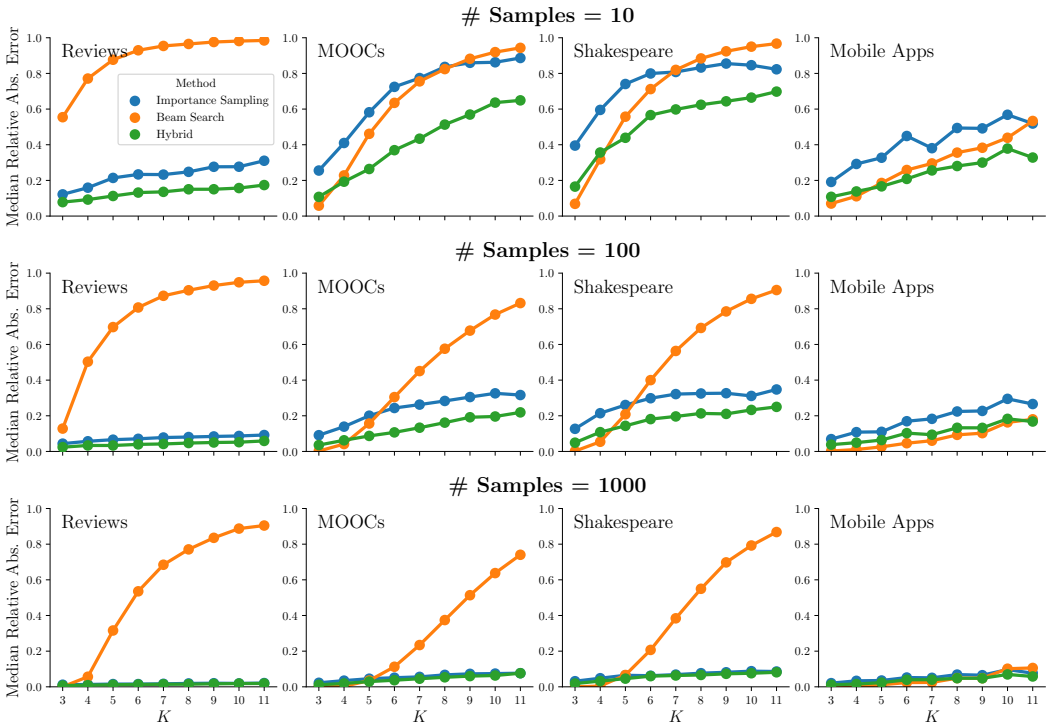

Figure 6: **Median estimation error versus query horizon** K: median relative absolute error (RAE) between estimated probability and (surrogate) ground truth for $p^*_\theta(\tau(\cdot) = K)$, for importance sampling, beam search, and the hybrid method, with varying computation budgets determined by 10, 100, and 1000 samples for the hybrid method. Ground truth values used to determine error in these plots are exact for $K \leq 4$ and approximated otherwise.

.

H.1 Query Estimation Error as a Function of Horizon K

Fig. 3 in the main paper shows the median relative absolute error (RAE) (across $N_Q = 1000$ queries) as a function of query horizon K for 4 datasets, comparing beam search, importance sampling, and the hybrid method, with a computation budget fixed at $S = 100$ hybrid samples. Here we provide a number of extensions of these results.

Fig. 6 shows the median RAE, for three levels of computation budget: $S = 10, 100, 1000$ and Fig. 7 shows the same results but now reporting mean RAE on the y-axis. While the details differ across different settings, the qualitative conclusions in these Figures agree with those for Fig. 3 in the main paper, namely that beam search is more sensitive (in its error) to both the horizon query K and to individual datasets, compared to both importance sampling and the hybrid method. More granular perspectives of this information for one of the datasets (MOOCs) can be seen in Figures 8 to 10 in the form of scatter plots of each of the individual query estimates against (surrogate) ground truth for that query. Different budgets are shown in different figures, and the results for $K = 3, 5, 7, 9, 11$ are shown in each column.

H.2 Query Estimation Error as a Function of Computation Budget

In addition to identifying optimal query estimation methodologies for a low and fixed computation budget, we also explore the impact of increasing computation budget for query lengths $K = 3, 7, 11$, roughly corresponding to short, medium, and long horizon queries. These experiments are conducted in the same manner as the query estimation experiments with a fixed model budget, but are then repeated for many different budgets derived from $S = 10, 30, 50, 100, 300, 500, 1000, 3000, 5000, 10000$ hybrid samples. The intention with these experiments is to observe if any query estimation methods disproportionately benefit from increased

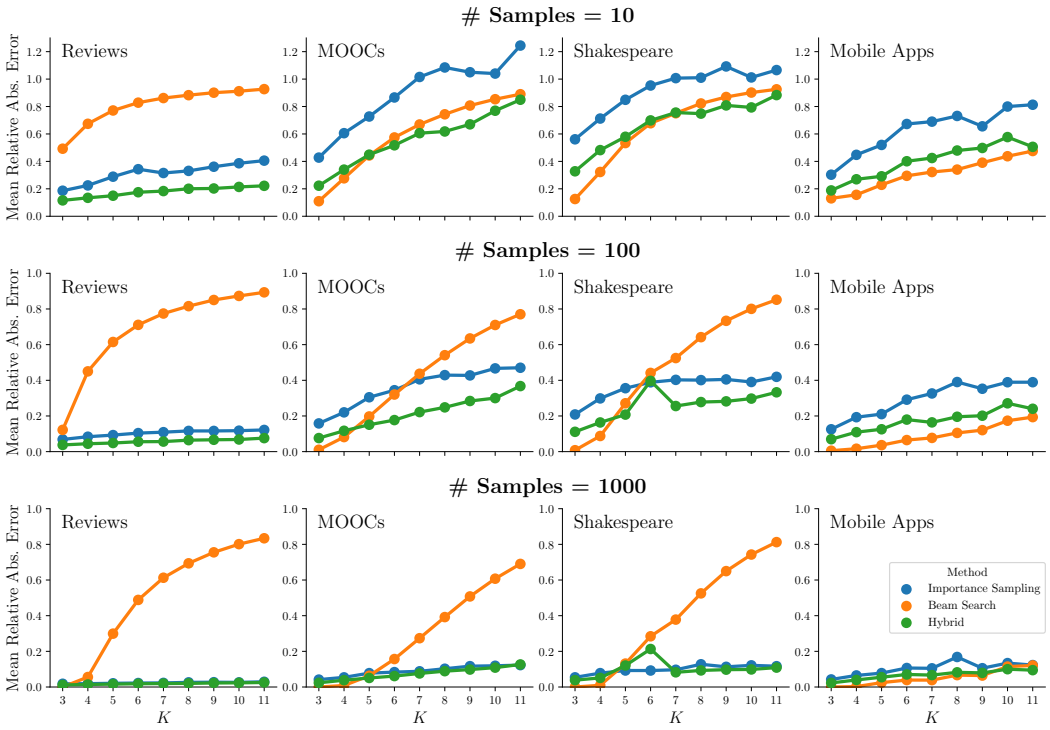

Figure 7: **Mean estimation error versus query horizon** K: same format as Fig. 6. Ground truth values used to determine error in these plots are exact for $K \leq 4$ and approximated otherwise.

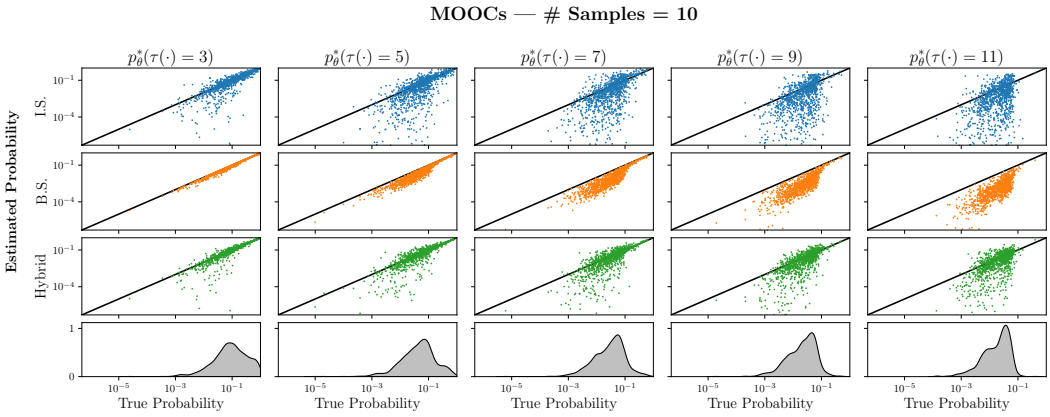

Figure 8: **Scatterplots of individual query estimates versus (surrogate) ground truth, computation budget of 10 hybrid samples:** Comparison of importance sampling (I.S.), beam search (B.S.), and the hybrid method for the MOOCs dataset with the budget determined by the hybrid method using 10 samples. The x-axis corresponds to the surrogate ground truth values for a given query result. Density plots at the bottom are for the surrogate ground truth values. Ground truth values used to determine error in these plots are exact for $K \leq 4$ and approximated otherwise.

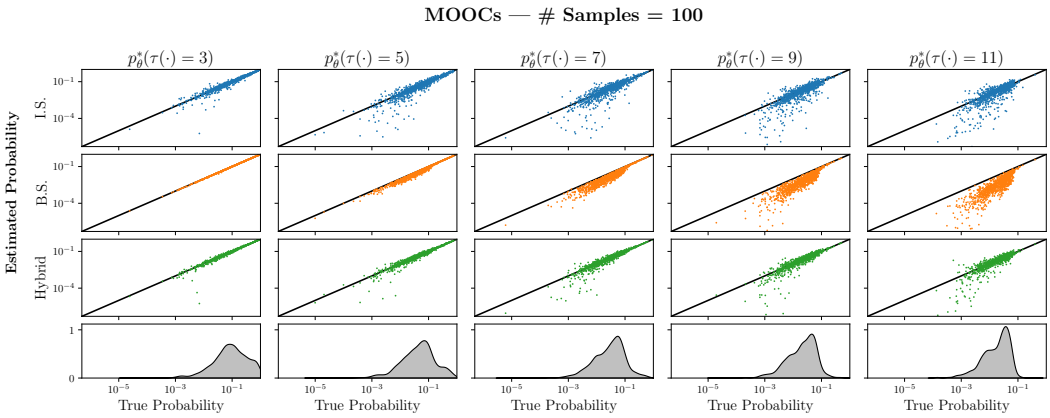

Figure 9: **Scatterplots of individual query estimates versus (surrogate) ground truth, computation budget of 100 hybrid samples:** Same format as Fig. 8.

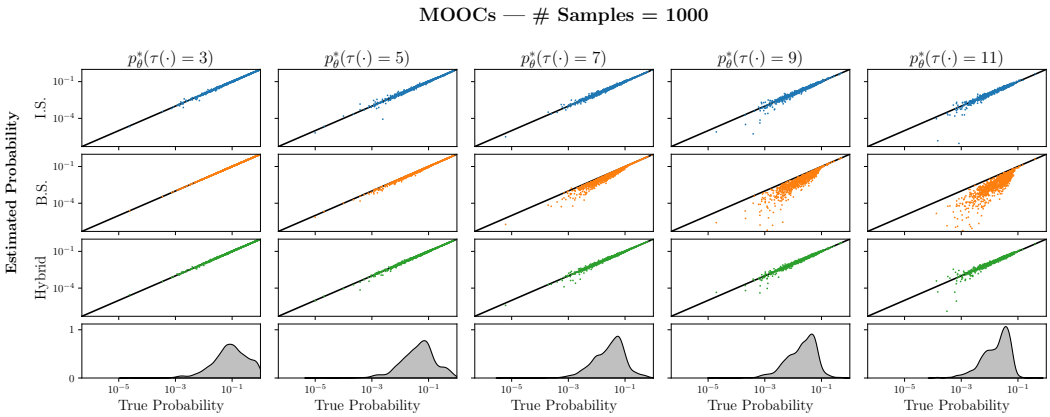

Figure 10: **Scatterplots of individual query estimates versus (surrogate) ground truth, computation budget of 1000 hybrid samples:** Same format as Fig. 8.

computation and exhibit behavior that was not present at lower computation budgets. We also include two additional baselines. The full set of methods explored is listed below.

1. Importance sampling (informative proposal distribution q derived from model p_θ)
2. Beam search
3. Hybrid search and sampling
4. Monte-Carlo sampling with a uniform proposal distribution
5. Naive model sampling (direct MC sampling for $\mathbb{E}_{p^*_\theta} \mathbb{1}(X_{1:K} \in \mathcal{Q})$)

As a clarifying point, naive sampling is conducted by sampling sequences from the model and determining if they fall within the query set $\mathcal{Q}$; the proportion of samples that exist in $\mathcal{Q}$ serves as a naive means of determining the query estimate in question. In addition, Monte-Carlo sampling with a uniform proposal distribution samples sequences in $\mathcal{Q}$ uniformly and the estimates the query probability for that sample. These two methods were not included in the main paper due to their consistently poor performance.

Fig. 11 (median error) and Fig. 12 (mean error), show that increasing the computation budget by an order of magnitude roughly corresponds to a three times reduction in RAE for both importance sampling and the hybrid method. Naive sampling sees almost no benefit from the increased budget regardless of the size of the query path space. Monte Carlo sampling sees some reduction in error from increased computation budget for some configurations, but also often sees no benefit.

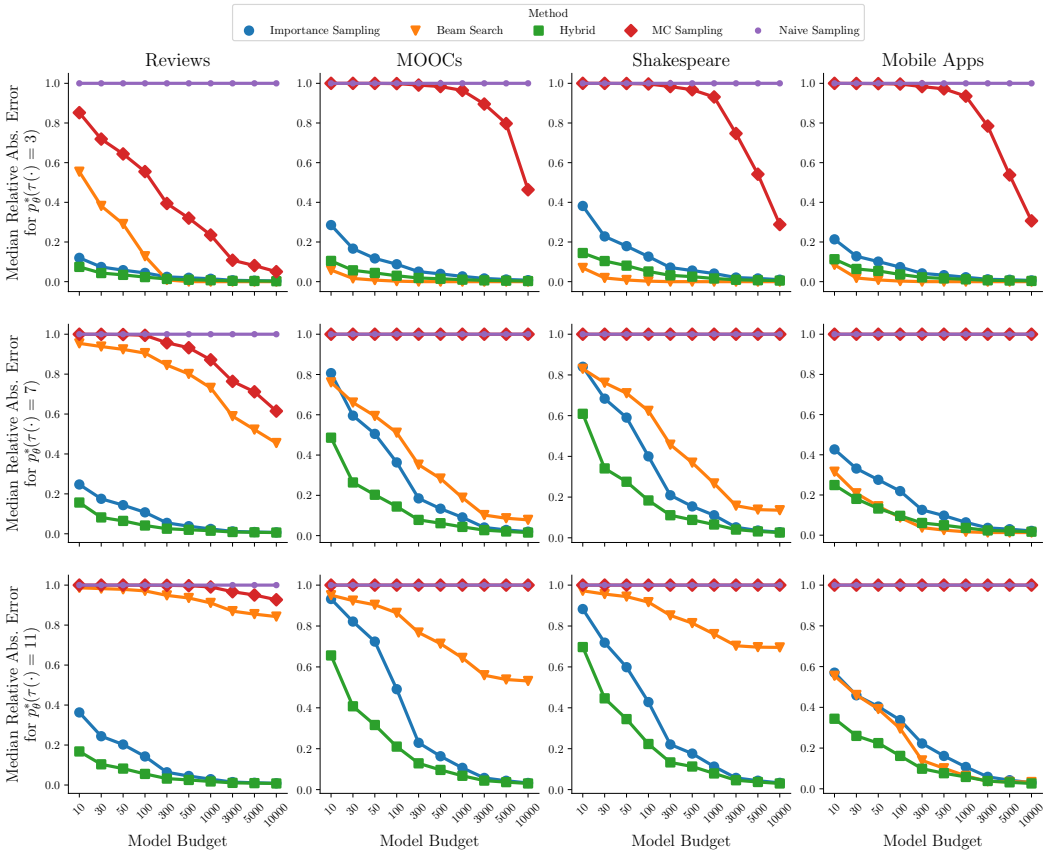

Figure 11: **Median estimation error versus model budgets:** Median relative absolute error (RAE) for importance sampling, beam search, the hybrid method, MC sampling, and naive sampling across three different queries, $p_\theta^*(\tau(\cdot) = k)$, for $k = 3, 7, 11$, over all four main datasets, as a function of different model budgets. For cases where the MC sampling results are not visible (e.g., see rightmost plot on the bottom row), the results coincide with the naive sampling results. Ground truth values used to determine error in these plots are exact for $K \leq 4$ and approximated otherwise.

For the provided budgets, the query estimates resulting from naive model sampling are consistently 0, resulting in an RAE of 1. While naive model sampling can be useful in some contexts in general, these results indicate that it is not well-suited for estimating query probabilities. This is likely because many of the ground truth probabilities for these queries have values on the order of 10^{-1} or smaller. For queries that are highly unlikely under the model, the probability of even a single sampled sequence belonging to $\mathcal{Q}$ is very low.

Monte-Carlo sampling also includes high error estimates, but for a different reason. Since the Monte-Carlo estimate can be decomposed into an expectation over $p(X_{1:K} = x_{1:K}, x_{1:K} \in \mathcal{Q})$ that is then re-scaled by $|\mathcal{Q}| \gg 0$, the scaling term magnifies any error in the expectation dramatically, inducing extremely high variance and the potential to produce query estimates that exceed 1. This high variance can persist even for high computation budgets. By contrast, beam search improves with an increased computational budget, but only as a function of the total path space. The larger the path space and the higher the entropy of the distribution, the worse the beam search estimates are as measured by RAE.

H.3 Coverage-based Beam Search Ablation

As described in the main paper, a variant of beam search well-suited to query estimation is *coverage-based beam search*. However, this method was not explored further in our analysis as it could not scale to non-trivial query types and large query path spaces. Depicted in Fig. 13, the minimum number

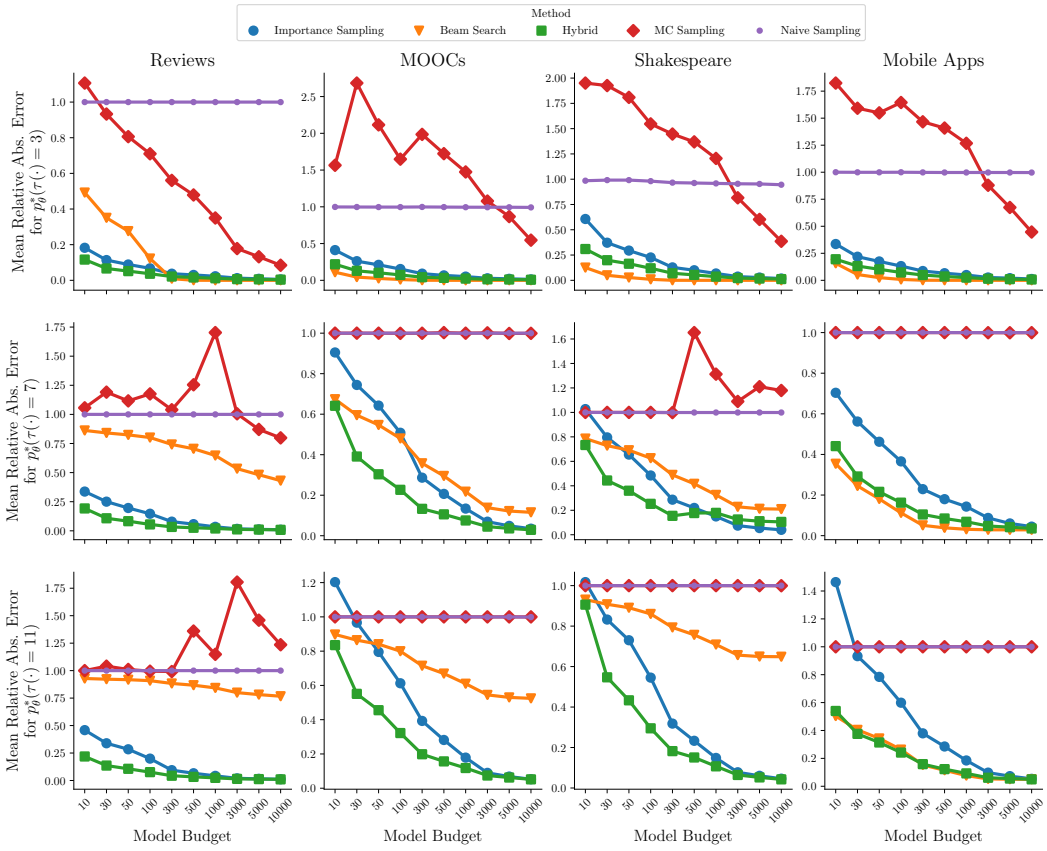

Figure 12: **Mean estimation error versus model budgets:** Same format as Fig. 11. Ground truth values used to determine error in these plots are exact for $K \leq 4$ and approximated otherwise.

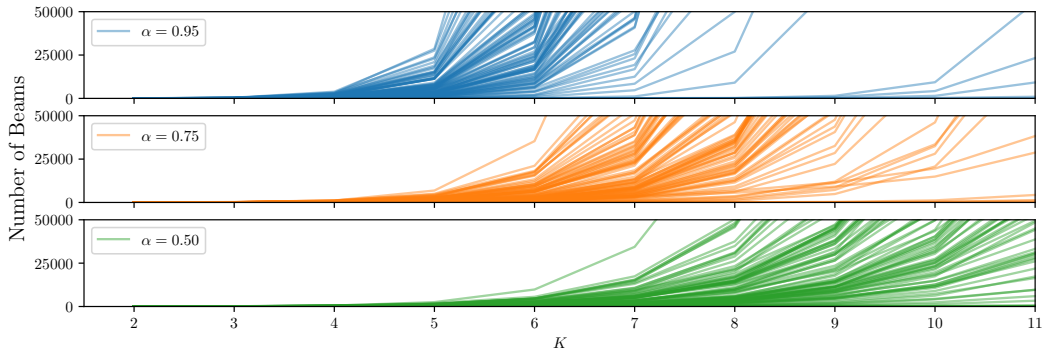

Figure 13: Number of beams as a function of K steps into decoding sequences for coverage-based beam search estimating $p_\theta^*(X_K)$ with $\alpha \in \{0.95, 0.75, 0.5\}$ over a variety of different starting histories in the Shakespeare dataset.

of beams needed to cover 50%, 75%, and 90% of the query path space increases exponentially with the query length K. Though coverage-based beam search comes with desirable coverage guarantees and is a consistent (albeit biased) query estimator, naively applying coverage-based beam search to queries of practical interest is ill-posed and computationally intractable. However, the hybrid method preserves some of beam search's desirable traits while making it suitable for queries of practical interest. See Appendix D for more information.

H.4 Long-horizon Query Estimation

The empirical results of our experiments tell us much of the abilit, but we still do not have clarity on the extent to which importance sampling can yield low-error estimates in the limit of large sequence lengths. To that end, we conducted an experiment to directly assess the performance of importance sampling when estimating long horizons. Our primary conclusion was that (i) with reasonable sample sizes the error remains below 25%, even at $K = 100$, and (ii) the error does not increase substantially as K increases beyond $K = 5$.

In more detail, a set of 100 history sequences $\mathcal{H}$ of length 5 was collected from each dataset. We then compute ground truth or pseudo ground truth (PGT) for query 2 (probability of event at K without restriction) over sequence lengths $K = [2, 5, 10, 20, 40, ..., 100]$. Pseudo-ground truth estimates are generated with a tolerance $\delta = 1e^{-7}$ and a maximum sampling budget of $250,000$. The same estimates are then computed using importance sampling with sampling budgets $[10, \ldots, 10000]$. Similar to the experiments in Section Section 5, we aggregate and present the results as the median relative absolute error, shown in the tables below. Since we are not focusing on a specific event type of interest, this median also marginalizes over all event types in the vocabulary as well as the sampled query sequences. All results are reported as percentages

# Samples	2	5	10	20	40	60	80	100
10	31.35	40.33	44.39	43.48	45.98	45.84	49.44	50.05
100	13.29	17.12	18.73	19.46	20.41	21.65	21.82	21.86
1000	4.94	6.38	7.06	7.33	7.26	6.99	7.13	6.82
10000	1.60	2.05	2.21	2.34	2.20	2.13	2.06	2.10

Table 2: Median RAE across query estimations methods (1000 samples) and query horizons $K = 2, 5, 10, 20, ..., 100$ and sample budgets $10, 100, 1000, 10000$ for Amazon Reviews.

# Samples	2	5	10	20	40	60	80	100
10	66.21	84.93	92.92	96.96	98.41	99.00	99.19	99.32
100	62.36	80.94	89.79	94.00	96.16	97.05	97.38	97.66
1000	47.96	59.14	69.10	75.45	74.79	71.34	65.12	59.75
10000	15.21	20.51	23.38	24.00	20.66	18.78	16.48	15.55

Table 3: Median RAE across query estimations methods (1000 samples) and query horizons $K = 2, 5, 10, 20, ..., 100$ and sample budgets $10, 100, 1000, 10000$ for Mobile Apps.

# Samples	2	5	10	20	40	60	80	100
10	79.84	79.99	83.53	85.29	83.53	84.00	85.82	85.15
100	28.66	32.85	39.25	40.66	38.78	41.51	39.71	40.07
1000	8.61	10.99	13.29	13.52	13.78	13.89	13.64	14.10
10000	2.77	3.48	4.28	4.34	4.32	4.33	4.34	4.30

Table 4: Median RAE across query estimations methods (1000 samples) and query horizons $K = 2, 5, 10, 20, ..., 100$ and sample budgets $10, 100, 1000, 10000$ for Shakespeare.

In general, we see (not surprisingly) that the increase in sequence length leads to a consistent and non-trivial increase in error for most sampling budgets. In addition, as expected the increase in sampling budget consistently reduces the query estimation error. However, we do witness the

# Samples	2	5	10	20	40	60	80	100
10	89.62	97.30	98.73	99.19	99.08	99.06	98.91	98.87
100	63.80	83.34	84.77	79.15	66.89	62.99	60.28	61.30
1000	30.39	43.37	36.32	26.07	19.78	18.11	17.53	17.38
10000	11.92	15.15	11.35	7.86	5.90	5.66	5.43	5.40

Table 5: Median RAE across query estimations methods (1000 samples) and query horizons $K = 2, 5, 10, 20, ..., 100$ and sample budgets $10, 100, 1000, 10000$ for MOOCs.

K	Importance Samp.	Beam Search	MC Samp.	Entropy	Entropy %
3	**11.41**	51.25	99.36	12.89	41.28
4	**13.35**	82.42	99.95	19.09	40.74
5	**13.53**	93.59	99.99	25.23	40.38

Table 6: **Median RAE across query estimations methods (1000 samples) and query horizons $K = 3, 4, 5$ for GPT-2 and Wikitext**. Entropy values estimated as the mean (over 100 queries) of the restricted proposal q and entropy % is the entropy in percentage relative to its potential maximum value $Klog(V)$.

interesting phenomenon that the error occasionally *decreases* as the sequence length increases. We conjecture that this may be happening because as the sequence length increases, the relevance of the history context $\mathcal{H}$ decreases and the distribution may regress to a base stationary distribution (as if no history context were provided at all) indicating that the conditional model entropy may be the main driving factor in estimation complexity. This intuition is further supported by the fact that in many datasets, budgets exist where query estimation error first increases but then begins to decrease again. Regardless, as the largest budget of $10,000$ we witness that median RAE remains at or under 25% in all cases, often significantly so.

H.5 Query Estimation with Large-Scale Language Models

In order to explore the feasibility of applying our query estimation methods to real-world sequence data, we also analyze a subset of our query estimation methods against GPT-2 and WikiText language data. GPT-2 decomposes English words into $V = 50257$ work pieces, a vocabulary over 500 times larger than our other datasets. For this reason, we only conduct experiments using only 100 sequence histories per dataset due to computational limitations. For the same reason, we do not explore the hybrid method and restrict ourselves to analyzing the following query estimators:

1. Importance sampling (informative proposal distribution q derived from model p_θ)
2. Beam search
3. Monte-Carlo sampling with a uniform proposal distribution

Fixed-budget query experiments with GPT-2 are conducted identically to those on the other 4 datasets. We find query estimation error closely mirrors the results we see in datasets with smaller vocabulary sizes, suggesting our findings may generalize well to practical domains. Our analysis is reported in Table 6 and includes estimates of the restricted model entropy $H(q)$ for different query lengths K, with the entropy increasing much faster than small-vocabulary models, as expected. With that said, there is still much exploration to be done on large-scale sequence models and is a promising avenue for future work.

H.6 Entropy Relationship with Query Estimation Error

Fig. 4(b) in the main paper demonstrated how indirectly controlling the entropy of a given model through an applied temperature affected the performance of beam search and importance sampling. In Fig. 14 we can see similar plots for all of four of our main datasets.

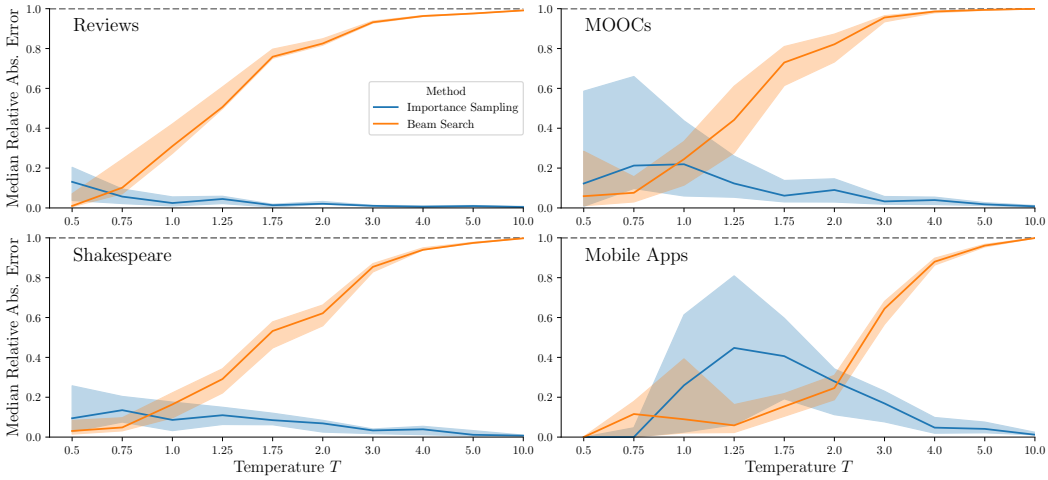

Figure 14: **Error as a function of entropy:** Median relative absolute error for estimating $p^*_{\theta,T}(\tau(\cdot) = 4)$ over all four datasets for a variety of imposed temperatures T. Shaded regions indicate interquartile ranges.

H.7 Investigation of Query 4 ("A" before "B")

Most of our analysis was conducted on hitting time queries, and justifiably so as more advanced queries decompose into individual operations on hitting times. This includes Q4, colloquially stated as the probability an item from token set A occurs before an item in token set B. More formally:

$$
\begin{aligned}
p^*_\theta(\tau(A) < \tau(B)) &= \sum_{k=1}^{\infty} p^*_\theta(\tau(A) = k, \tau(B) > k) \\
&= \sum_{k=1}^{\infty} \sum_{a \in A} p^*_\theta(X_k = a, X_{<k} \in (\mathbb{V} \setminus (A \cup B))^{k-1})
\end{aligned}
$$

While this cannot be computed exactly, a lower bound can. The other option is to produce a lower bound on this expression by evaluating the sum in Eq. (5) for the first K terms. We can achieve error bounds on this estimate by noting that $p^*_\theta(\tau(a) < \tau(b)) + p^*_\theta(\tau(a) > \tau(b)) = 1$. As such, if we evaluate Eq. (5) up to K terms for both $p^*_\theta(\tau(a) < \tau(b))$ and $p^*_\theta(\tau(b) < \tau(a))$, the difference between the sums will be the maximum error either lower bound can have. This difference will be referred to as *unaccounted probability* and will approach 0 as $K \to \infty$.

A natural question to ask is what is the minimum value of K sufficient to compute these lower bounds to in order to have negligible unaccounted probability. Though this will surely vary based on the entropy of the model and the specific query in question, we explore this question across all datasets except WikiText and note some general trends.

Fig. 15 plots the unaccounted probability $1 - \hat{p}^*_\theta(\tau(A) < \tau(B)) - \hat{p}^*_\theta(\tau(A) > \tau(B))$ as a function of query length K. We observe that for many datasets, a query horizon of $k = 30$ is largely sufficient to reduce the remaining probability to under 10%. One notable exception is the Mobile Apps dataset, which, due to its lower entropy and high self-transition rate, maintains a much longer query horizon. This discovery implies that a successful partition of probability space with a given Q4 query can be sensitive to the model distribution, but also that approximate partitions are possible for relatively low values of K.

H.8 Linearly Compounding Errors in Complex Query Estimation

As mentioned in Section Section 5 of the main paper, hitting time queries can often be seen as components of more involved queries, such as "a" before "b" queries $p^*_\theta(\tau(a) < \tau(b))$ or counting-style queries $p^*_\theta(N_a(K) = n)$. These queries can be rewritten as summations of more basic hitting time queries. For instance, $p^*_\theta(\tau(a) < \tau(b)) = \sum_{k=1}^{\infty} p^*_\theta(\tau(a) = k, \tau(b) > k)$. In practice, each

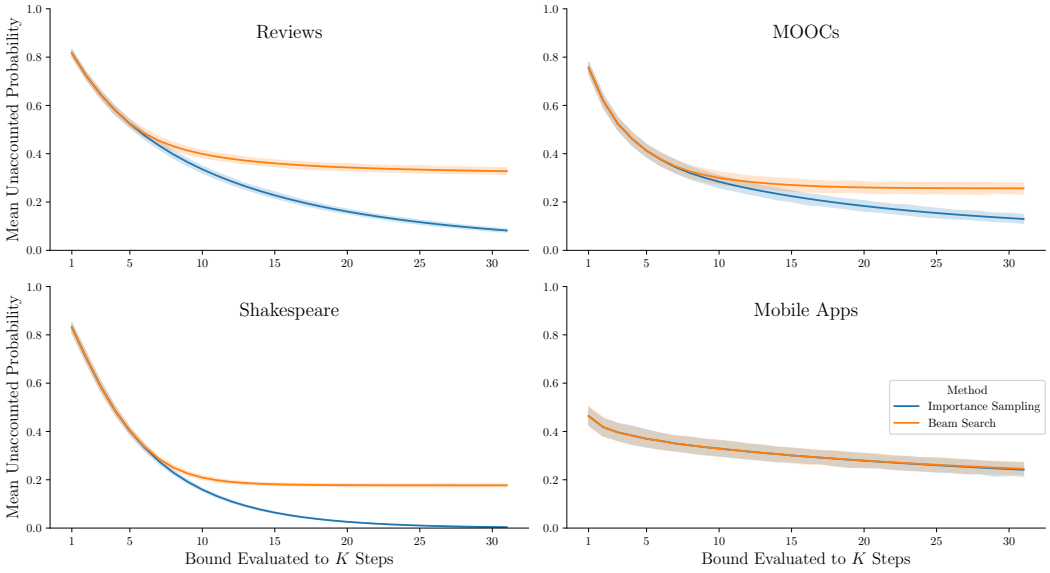

Figure 15: Mean unaccounted probability $(1 - (\hat{p}^*_\theta(\tau(A) < \tau(B)) + \hat{p}^*_\theta(\tau(B) < \tau(A))))$ when evaluating the pair of queries for $\tau(A) < \tau(B)$ and $\tau(B) < \tau(A)$ up to K steps into the future over all four main datasets. Shaded regions indicate 99% confidence intervals and estimates were computed with a fixed model budget based on 1000 samples.

summand probability is estimated using our proposed techniques so it can be seen that the error compounds additively with respect to the different basic hitting time queries.

More generally, our framework proposes representing general queries as $\mathcal{Q} = \cup_i \mathcal{Q}_i = \cup_i \prod_j \mathcal{V}_j^{(i)}$ such that the $\mathcal{Q}_i$'s form a minimal partition on $\mathcal{Q}$. With this representation, we can see that for an arbitrary query, the error when estimating will compound additively and scale linearly with respect to the number of different $\mathcal{Q}_i$. Note that the actual values of $p^*_\theta(x \in \mathcal{Q}_i)$ do have an impact on the errors when estimating due to the values $\in [0, 1]$. Lastly, as mentioned in the paper we can additionally control this error either by utilizing the coverage-based beam search, or by leveraging the Central Limit Theorem with importance sampling.

H.9 Qualitative Exploration and Practical Applications

In addition to systematic quantitative analysis of query estimators, we also qualitatively explore specific applications of our methods that lend practical insights. First, we consider the question of "given a partial sentence, predict when the sentence will end". Using our query estimation methods, we can not only answer this question with relatively low error but also effectively re-use intermediate computational results. This necessarily correlates query estimates over steps K, but this confers little negative impact upon the analysis since our sampling methods are unbiased estimators and beam search, though biased, offers a deterministic lower bound. The results of our analysis are seen in Fig. 1 in the main paper and align with basic intuition about English sentences, with open ended prefixes possessing a long-tailed end-of-sentence probability distribution relative to more structured and declarative phrases.

A second practical question that can be asked with our query estimation methods utilizes the Mobile Apps dataset: given someone's mobile usage history, predict what will occur first: the individual will go on social media or directly interact with someone via video chat, call, text, or email. This question is a practical application of query 4, $p^*_\theta(\tau(a) < \tau(b))$. Other, equally interesting equivalents of this question include "will an online shopper purchase something before leaving the website?". Below, we have conducted an experiment where we synthetically generate mobile app behavior histories with specific percentages of social media activity present. By computing these queries over several histories and then averaging the estimates, we obtain the results seen in Fig. 16. As expected, we see a clear linear trend between a user's social media usage and the likelihood they will return to

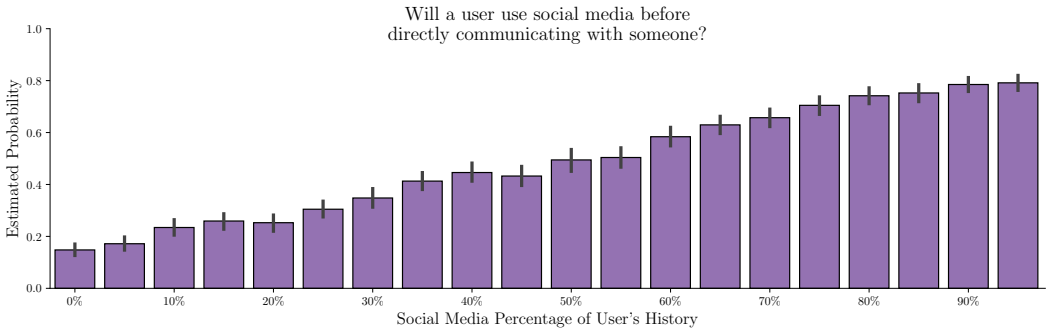

Figure 16: A case study of evaluating how likely it is that a mobile app user will use social media before they directly communicate with someone via text messaging, email, phone call, etc. as a function of how much of their recent history consists of social media usage. This is a $p^*_\theta(\tau(A) < \tau(B))$ type query that has been estimated and averaged over 500 different histories for each social media history percentage. Error bars indicate 90% confidence intervals.

it before conducting other tasks like directly communication. Such a result, though contrived and purely demonstrative in this setting, could be applied to many practical applications that analyze the characteristics of online user behavior.