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

* Denotes equal contribution {alexjb, showalte}@uci.edu

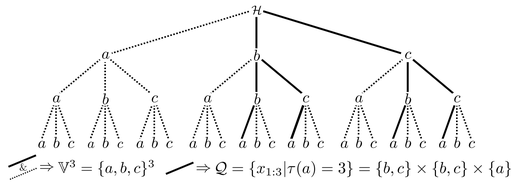
Selected as Oral

Paper (ArXiv)

Abstract

- Autoregressive models capture detailed information about future events.
- To-date, investigation is centered solely on next-event prediction.
- **Goal:** Formalize complex queries such as "event A occurs before B"
- **Goal:** Leverage our formalism to produce methods for estimating complex queries that exist in exponentially large sequence spaces

Introduction



$X_{1:K}$ - Sequence of K random variables
$\mathbb{V}$ - Discrete, fixed-size vocabulary
$\mathscr{H}$ - Observed sequence history $x_{1:k}$
$\mathcal{Q}$ - Query space for $X_{1:K}$ where $\mathcal{Q} \subset \mathbb{V}^K$
p_θ - Learned probability distribution
$p^*_\theta(\cdot)$ - Shorthand for $p_\theta(\cdot \mid \mathscr{H})$
$\mathbf{1}$ - Indicator function
K - Query length

- A query $\mathcal{Q}$ is defined as a subspace of sequence vocabulary $\mathbb{V}^K$

$$\mathcal{Q} = \cup_i \mathcal{Q}^{(i)} \text{ where } \mathcal{Q}^{(i)} \cap \mathcal{Q}^{(j)} = \varnothing \text{ for } i \neq j$$
$$\text{and } \mathcal{Q}^{(i)} = \mathscr{V}^{(i)}_1 \times \mathscr{V}^{(i)}_2 \times \ldots \times \mathscr{V}^{(i)}_K \quad \text{where } \mathscr{V}^{(i)}_k \subseteq \mathbb{V} \text{ for } k = 1,\ldots,K$$

- We seek to approximate queries of interest with the following

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

- With this query formalism, we can decompose this process:

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

- We seek other estimation methods beyond Monte-Carlo sampling:

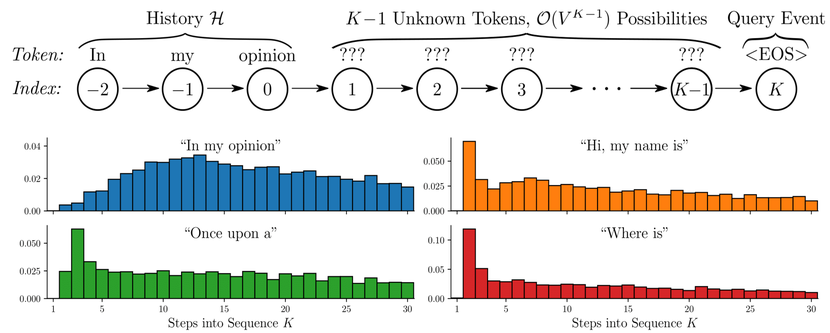
$$p^*_\theta(X_{1:K} \in \mathcal{Q}) = \mathbb{E}_{x_{1:K} \sim p^*_\theta}\left[\mathbf{1}(x_{1:K} \in \mathcal{Q})\right]$$

Probabilistic Queries

#	Question	Probabilistic Query	Cost $(K \cdot \lvert\mathcal{Q}\rvert)$
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?	$p^*_\theta(N_a(K) = n)$	$\mathcal{O}\left(\binom{K}{n}(V-1)^{K-n}\right)$

- There are many queries of interest in sequence modeling, but all require marginalization over *exponentially large path spaces*

Probabilistic Queries Visualized



(**top**) Illustration of a query for the probability of a given sentence "In my opinion..." ending in K steps. (**bottom**) GPT-2 I.S. estimates of 4 prefixes ($|\mathbb{V}| = 50{,}257$). Relative efficiency over naive sampling is 6+.

Proposal Distribution

- Query estimation can be simplified with an autoregressive proposal distribution $q^{(i)}(\cdot)$ whose domain matches that of subquery $\mathcal{Q}^{(i)}$

$$q^{(i)}(X_{1:K} = x_{1:K}) = \prod_{k=1}^{K} p^*_\theta\left(X_k = x_k \mid X_{<k} = x_{<k}, X_k \in \mathscr{V}^{(i)}_k\right)$$
$$= \prod_{k=1}^{K} \frac{p^*_\theta\left(X_k = x_k \mid X_{<k} = x_{<k}\right)\mathbf{1}\left(x_k \in \mathscr{V}^{(i)}_k\right)}{\sum_{v \in \mathscr{V}^{(i)}_k} p^*_\theta(X_k = v \mid X_{<k} = x_{<k})}$$

Query Estimation Methods

- **Importance sampling** is a natural candidate to leverage proposal q

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

- **Beam search** can *guarantee coverage* by finding beams $\mathscr{B} \subset \mathcal{Q}$

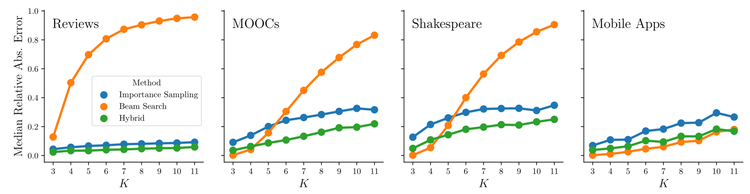
$$p^*_\theta(X_{1:K} \in \mathcal{Q}) = \sum_{x_{1:K} \in \mathcal{Q}} p^*_\theta(X_{1:K} = x_{1:K}) \geq \sum_{x_{1:K} \in \mathscr{B}} p^*_\theta(X_{1:K} = x_{1:K})$$

- Our novel **hybrid method** merges search and sampling by noting that:

$$p^*_\theta(X_{1:K} \in \mathcal{Q}) = \textstyle\sum_{x_{1:K} \in \mathscr{B}_K} p^*_\theta(X_{1:K} = x_{1:K}) + \sum_{x_{1:K} \in \mathcal{Q} \setminus \mathscr{B}_K} p^*_\theta(X_{1:K} = x_{1:K})$$

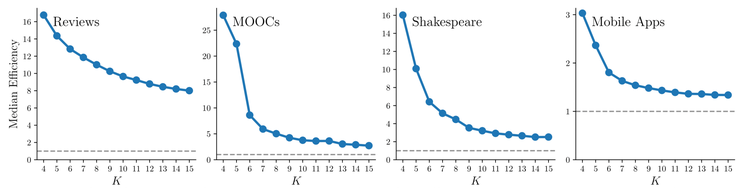
- Where $\mathscr{B}_K$ is the sequence set from beam search. We first search for likely sequences and then sample from the remaining space with $q(X_{1:K} = x_{1:K} \mid X_{1:K} \notin \mathscr{B}_K)$
- Ground-truth is often *intractable* to compute. We evaluate our methods against pseudo-ground truth (PGT) estimates generated with importance sampling
- PGT leverages a high compute budget and the convergence guarantees of the CLT

Query Estimation Error



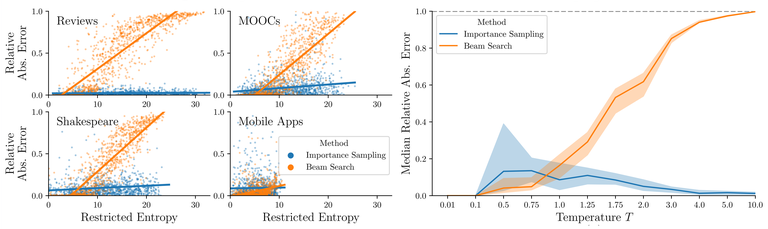
Median relative absolute error (RAE) for for hitting-time query estimates over 4 datasets and 3 estimation methods. Hybrid method outperforms others including naive sampling.

Relative Efficiency



Median rel. efficiency of importance sampling for all datasets. For the regime of K where computing ground truth is intractable, we see a significant boost over naive sampling.

Query Estimation and Entropy



(**left**) Median relative absolute error (RAE) vs restricted entropy per query (with best linear fits), (**right**) Median RAE v. model temperature T for Mobile Apps data. Beam search error highly correlates with entropy, and increasing T induces 100% error.

Composability and Saving Computation

- More complex queries (e.g. $\mathbf{Q}4$, $\mathbf{Q}5$) can be decomposed into operations over hitting-time queries ($\mathbf{Q}3$)

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

- For this reason, experiments focus on hitting-time queries
- One can save computation between queries $\mathbf{Q}$ and $\mathbf{Q}'$ by re-using sub-sequences of $p^*_\theta(X_{1:K} \in \mathcal{Q})$ on $\mathcal{Q}'$ if $\mathscr{V}_i = \mathscr{V}'_i$ for $i = 1, \ \ldots \ , K-1$