Optimizing Importance Sampling Methods for Rare Output Estimation in LLMs

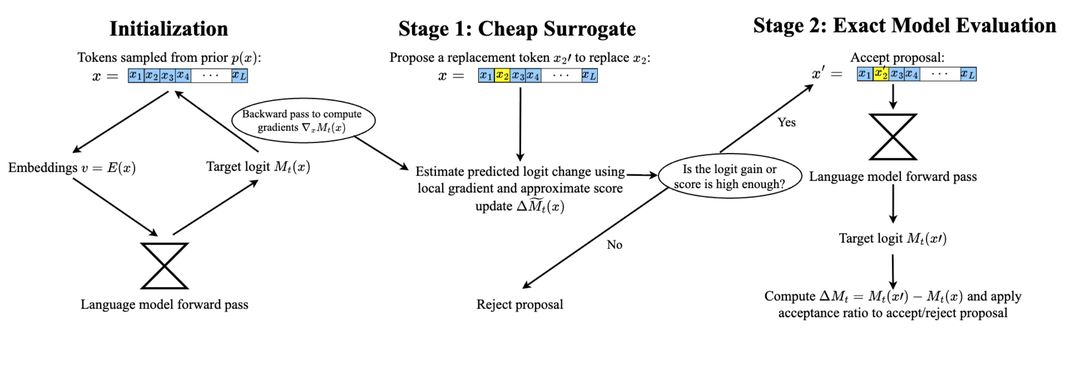
Amanda Cao[1, 2], Ivan Betancourt[1, 3], Manish Rangan[1, 4], Yuqi Sun[1, 5]

[1]*Algoverse AI Research Program,*
[2] *Yale University, Department of Statistics & Data Science, New Haven, USA*
[3]*Amherst College, Department of Computer Science, Amherst, USA*
[4]*Georgia Institute of Technology, Department of Computer Science, Atlanta, USA*
[5]*Mindoverflow Co.*

Introduction

Large language models (LLMs) may produce rare but potentially catastrophic outputs, especially in deployment settings where models could experience distribution shift. **Understanding a model's worst-case performance** is necessary to ensuring the safety and reliability of artificial intelligence systems. However, because rare outputs have true probabilities as low as 10^{-9}, standard sampling methods are computationally unfeasible. Previous works have introduced **Metropolis-Hastings Important Sampling (MHIS)**, an algorithm that samples from a Boltzmann-weighted distribution to better explore rare-event regions in the input space. Here, we introduce **Delayed-Acceptance Metropolis-Hastings Importance Sampling (DA-MHIS**), an algorithm that **optimizes MHIS for efficiency by filtering out poor-quality proposals**. We also mention Multiple-Try Metropolis Hastings Importance Sampling as an additional variant.

Methodology



This paper builds on the **Metropolis-Hastings Importance Sampling** (MHIS) algorithm outlined by Wu & Hilton (2025). MHIS attempts to estimate the probability that a model M outputs a specific target token t over a distribution of input tokens by constructing a Markov chain that up-samples inputs with a high likelihood of producing token t.

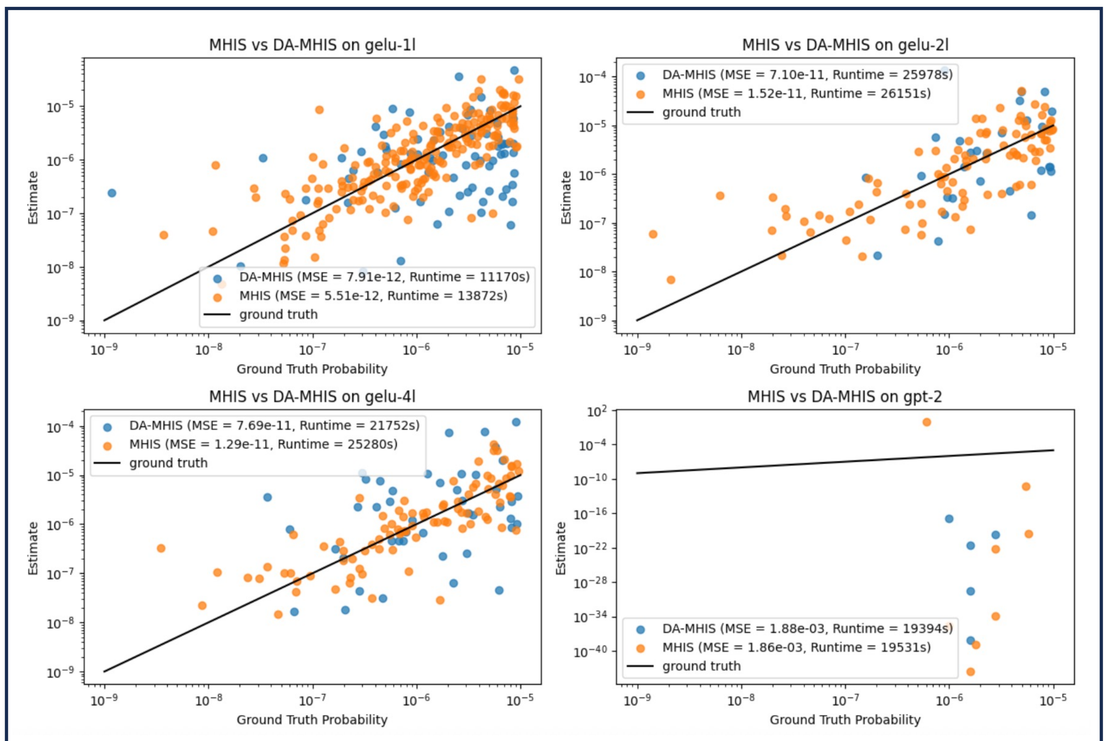
However, MHIS and related algorithms are computationally expensive, as each proposal requires a full forward and backward pass to evaluate the acceptance ratio.

Our work introduces the **Delayed-Acceptance Metropolis-Hastings Importance Sampling** method:

1. Sample an initial input from the original distribution
2. Run the chain over a specified number of iterations, where each iteration proposes an edit to the current input
3. Randomly select a token position to update
4. Create a surrogate proposal token
5. Screen the proposal token with a lightweight, gradient-based test that discards unsatisfactory proposals before model evaluation
6. Construct the proposed new input and compute the Metropolis-Hastings acceptance ratio.

This two-stage mechanism allows the estimator to concentrate computation on high-value proposals, discarding “useless” candidates.

Results



To evaluate the effectiveness of the proposed algorithms, we tested DA-MHIS against the original MHIS algorithm. Initial experiments were conducted on a GELU 1-layer model, and scaled up to GELU 2-layer, GELU 4-layer, and gpt-2 models, respectively. Experiments were conducted on a representative sample of $n = 350$ ground truths.

We find that on the GELU 1-layer model, **DA-MHIS reduces runtime by 16.4% as compared to the baseline**, with runtime benefits observed for GELU 2-layer and GELU 4-layer models. Comparing the MSEs, we also note that DA-MHIS maintains comparable accuracy as compared to MHIS.

However, when scaling up to GPT-2, the performance of both methods deviated significantly from the ground truth. We speculate that the GPT-2 estimates skew low because in the 10^{-9} regime and with a fixed 2^{16}-call budget, effective sample sizes are tiny and the chain rarely visits the rare-event region.

Conclusion and Future Works

We identify Delayed-Acceptance Metropolis-Hastings Importance Sampling as a promising method for optimizing the efficiency of the Metropolis Hastings Importance Sampling algorithm. DA-MHIS delivers substantial improvements upon runtime without sacrificing low-probability estimation accuracy.

We would be excited about **further research on scaling such optimization methods to larger models**, through methods such as examining the gradient descent processes to determine whether current efficiency gains scale consistently with model size.